University College London

Department of Electronic & Electrical Engineering

Preserving individual privacy in context-aware ubiquitous computing environments —— An intelligent and distributed agent technology for context-dependent privacy control

by Ni (Jenny) ZHANG

Supervisor: Professor Chris Todd

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Faculty of Engineering of the University of London, and for the Diploma of the University College London, 2008

London, United Kingdom

May 2008

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Declaration

I, Ni (Jenny) ZHANG, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Ni (Jenny) ZHANG

Abstract

Context-aware computing aims to take advantage of contextual knowledge to make decisions about how to dynamically provide services or adapt to meet user requirements. A tradeoff exists between preserving individual privacy and disclosing information to benefit from rich and interesting services. Although privacy issues have been recognized as a great barrier to the adoption and a long-term success of the context-aware computing, an extensive literature review conducted by the author has indicated that only a small subset of the privacy needs and challenges have been moderately addressed, and demand for adequate privacy protection in the context-aware paradigm is significant.

This doctoral work introduces a distributed privacy protection model to tackle the challenges and overcome the limitations of existing solutions, and proposes an intelligent agent technology to facilitate a relatively unobtrusive user participation in controlling the disclosure of their sensitive information. It aims at addressing two key concerns of preserving privacy in context-aware ubiquitous computing environments: privacy feedback (i.e. notifying individuals of relevant information disclosure) and privacy management (i.e. allowing individuals to express their privacy preferences and manage their privacy levels).

The proposal of the intelligent privacy agent is characterized by developing automated privacy preference mechanisms to enforce privacy control in response to context changes. More specifically, the author has developed a Privacy Policy/Preference Language to facilitate a common understanding of privacy requirments, and has exploited ontology-based methods to enable semantic policy analysis of context-dependent privacy preferences.

A proof-of-concept implementation using Web Service technologies demonstrates that the proposed privacy solution can be deployed to achieve interoperability across system platforms and devices. and is scalable to the global Internet. Quantatitive performance evaluations are conducted to validate the novel approaches of using hybrid reasoning mechanisms to perform the task of semantic privacy policy evaluation, preference conflict and redundancy detection, and context perception.

Acknowledgement

My first and foremost thanks go to my supervisor, Professor Chris Todd, Emeritus London University Professor of Network Science. I feel extremely luck to have Prof. Todd as my supervisor, who is always dedicated himself to the success of his students. I would like to sincerely thank him for his consistent and invaluable guidance and support during my entire UK study life, including both MRes and PhD researches. Prof. Todd has provided me with a perfect balance between guidance and freedom, which allows me to pursue my own ideas along the right direction.

I would like to extend my special thanks to Professor Alex Galis of the Networks and Services Group in E&EE-UCL. I am grateful to him for introducing me into the European Union (EU) Ambient Networks Project, by which I embarked on context-aware computing research and was inspired to look into personal privacy issues. I would also like to thank Dr. Miguel Rio of E&EE-UCL for his helpful discussions on the ideas in my transfer thesis, and other 19 external scholars in ubiquitous computing community for their critical comments to help improve this research work.

My thanks also goes to current and previous members of the Networks and Services Group of E&EE-UCL, especially Dr. David Griffin, Dr. Kun Yang, Dr. Jason Spencer, Dr. Alvin Tan, Dr. Roel Ocampo, Dr. Lawrence Cheng, Dr. Jonas Griem, Dr. Kerry Jean, Hamed Haddadi, Moses Woldeselassie, and Venus Shum. I thank them for sharing their PhD experience and research with me, and for their moral support during my PhD study.

Last but not the least, my most profound gratitude goes to my family. I am indebted to my parents for their greatest love, support and encouragement. Great gratitude goes to my beloved husband, Lijun Tang, who consistently supports me in pursuit of the

success in my research, and to our lovely baby, Xinyi TANG, whose delivery brings me joy and excitement. I owe my today's achievement to them and this dissertation is dedicated to them.

Table of Content

List of Figures

List of Tables

List of Abbreviations

ACAI	Agent-Based Context-Aware Infrastructure
ACL	Access Control List
APPEL	A P3P Preference Exchange Language
API	Application Programming Interface
AN	Ambient Networks
BDI	Belief-Desire-Intention
CAMUS	Context-Aware Middleware for Ubiquitous Computing Systems
CC/PP	Composite Capabilities/ Preference Profiles
CIM	Common Information Model
CoBrA	Context Broker Architecture
CONFAB	CONtext FABbric Toolkit
CONON	Context Ontology
CoOL	Context ontology language
CoPS	Context Privacy Service
CORBA	Common Object Request Broker Architecture
CPU	Central Processing Unit
CR-RBAC	Context, Rule and Role-Based Access Control
CSCP	Comprehensive Structured Context Profiles
DAML	DARPA Agent Markup Language
DL	Description Logic
DRBAC	Dynamic Role Based Access Control
DRM	Digital Rights Management
DOS	Denial Of Service
EC	European Commission
EPAL	Enterprise Privacy Authorization Language
ETH Zurich	Swiss Federal Institute of Technology in Zurich
EU	European Union
F-Logic	Frame Logic
FOAF	Friends-of-A-Friends
FTP	File Transfer Protocol
Geopriv	Geographic Location and Privacy
GPS	Global Positioning System
GRBAC	Generalized Role Based Access Control
HCI	Human Computer Interface
HTTP	HyperText Transfer Protocol
HP	Hewlett-Packard
IBM	International Business Machine
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPsec	Internet Protocol Security Protocol

ISO	International Standard Organization
IST	Information Society Technologies
MIME	Multipurpose Internet Mail Extensions
MoCA	Context-Provisioning Middleware
MORI	Market & Opinion Research International
OASIS	Organization for the Advancement of Structured Information Standards
OECD	Organization for Economic Cooperation and Development
OO	Object-oriented
OpenGIS	Open Geography Information System
OSI	Open System Interconnection
OWL	Web Ontology Language
OWL-DL	Web Ontology sublanguage-Description Logic
PA	Privacy Agent
PACE	Pervasive Autonomic Context-aware Environments
Parc	Xerox's Palo Alto Research Center
PawS	Privacy awareness System
PawDB	Private-aware Database
PDA	Personal Digital Assistant
PKI	Public Key Infrastructure
P3P	Platform for Privacy Preference
PPAP	Privacy Policy Administration Point
PPDP	Privacy Policy Decision Point
PPEP	Privacy Policy Enforcement Point
RBAC	Role-based Access Control
RDF	Resource Description Framework
RDF-S	Resource Description Framework Schema
RFC	Request for Comments
RMI	Remote Method Invocation
RPC	Remote Procedure Call
RSA	an algorithm for public-key encryption developed by by Ron Rivest, Adi Shamir and Len Adleman at MIT
RuleML	Rule Markup Language
SAML	Security Assertion Markup Language
SDSI	Simple Distributed Security Infrastructure
SOAP	Simple Object Access Protocol
SOUPA	Standard Ontology for Ubiquitous and Pervasive Applications
SPARQL	SPARQL Protocol and RDF Query Language
SWRL	Semantic Web Rule Language
SGML	Standard Generic Markup Language
SPKI	Simple Public Key Infrastructure
SSL	Secure Sockets Layer
TLS	Transport Layer Security
UML	Unified Modeling Language
URL	Uniform Resource Locator

VPN	Virtual Private Network
XACML	eXtensible Access Control Mark-up Language
XML	eXtensible Mark-up Language
XML-DSIG	eXtensible Mark-up Language –Digital SIGnature
Xpath	XML Path Language
Xpref	Xpath-based Preference Language for P3P
XSLT	eXtensible Stylesheet Language Transformations
W3C	World Wide Web Consortium
WASP	Web-Architecture for Service Platform
WS	Web Service
3G	Third Generation (of mobile communication systems)

Chapter 1: Introduction

1.1 Motivation

Since Weiser initiated the research on ubiquitous computing at Xerox Palo Alto Research Center (Xerox Parc) in 1991, context and context-awareness have been receiving increasing interest and has become a key driver in the ubiquitous computing paradigm [1]. The ubiquitous computing idea by Weiser was that computer systems should become invisible to users and disappear from conscious thought, and users should not be distracted any more from their tasks by concentrating on a particular computer interface [2].

Under such circumstance, it could be imagined that ubiquitous sensing and the invisible form of embedded computing devices will make it easier than ever to collect and use information about individuals without their knowledge. Sensitive private information might live indefinitely and appear anywhere at anytime. "It will become hard to know what is controlling what, what is connected to what, where information is flowing, how it is being used...and what are the consequences of any given action" [3]. As a result, there have been numerous interviews (e.g. [4, 5, 6]), essays (e.g. [3, 7]), books (e.g. [8, 9]), and repeated negative media coverage (e.g. [10,11]) describing people's concerns about strong potentials for abuse, unease over a potential lack of control, and general desire for privacy-respecting ubiquitous computing systems. These concerns suggest that although ubiquitous computing holds great promise, privacy issues may be the greatest barrier to its long-term success.

The privacy problems haunting ubiquitous computing will only worsen in the context-aware paradigm. Context-aware computing relies on various types of context information in order to make decisions about how to dynamically adapt to meet user requirements. This information is heterogeneous and is usually derived from a range

of sources (including user profiles, applications, sensors, etc.), with variable privacy requirements resulting from differences in the sensitivity of the information, differences in users' individual privacy preferences, and changes in users' privacy preferences over time and in response to context changes. The point is that, privacy in the context-aware paradigm is no longer a single monolithic concept, rather it is a fluid and malleable notion with a range of needs and trust levels, which makes it extremely challenging to provide adequate protection for sensitive context information, and to design and implement privacy solutions for context-aware systems.

As a result of the difficulties, most current context-aware systems provide very little support for privacy, although researchers often noted the importance of privacy and security in context-aware computing [12, 13, 14]. Most of the early efforts to address privacy concerns have been concerned with the design of privacy-preserving location sensing systems [15] and the integration of access control mechanisms into ubiquitous computing infrastructure [16, 17]. These solutions addressed only a small subset of the privacy needs and challenges faced in context-aware environments, and are based on many simplifying assumptions (e.g., they only consider location information, or assume that context information is neatly partitioned into repositories that are under the control of a single user)[1].

These limitations have driven the author's research to investigate potential approaches and mechanisms to address the challenges and overcome the limitations of existing solutions, and to work out a technical solution that is expected to be able to work with today's social and legal reality to offer adequate privacy protection for individuals to benefit from the advancement of context-aware intelligence.

[1] An extensive literature review conducted by the author on existing approaches and solutions has indicated limitations in approaches taken to date. This will be presented in Chapter 3.

1.2 Research scope

As suggested by the title of this doctoral thesis, the research work presented in this doctoral thesis is scoped by two key words: *Personal Privacy* and *Context-Aware Computing*.

Personal privacy is the focus of this work although some of the security mechanisms proposed in this work could offer protection to other entities involved in context-aware ubiquitous computing, for example, network routers. Since human users are the focus of privacy protection in this work, the author explores a privacy protection solution from a user-centric point of view. The solution proposed by the author aims at protecting individuals from unwanted information disclosure, and helping them manage their privacy requirements in a dynamic context-aware environment with relative ease. This is distinguished from conventional system-centric approaches that are mainly concerned with implementing access control and security mechanisms to protect information that is kept within their systems from unauthorized access or malicious attack. In the author's viewpoint, security and privacy are interdependent concepts, but they present distinct business and social issues. Security provides, in part, the means to achieve privacy, but it does not equal privacy protection that requires responsible and respectful acquisition and management of personal data. Security mechanisms are not sufficient to safeguard against subsequent use (once data is disclosed), to minimize the risk of sensor-based disclosure, and to reassure users. In a ubiquitous computing environment, privacy and security may be contradictory objectives. The enforcement of system security mechanisms may conflict with the task of preserving individual privacy if, for example, access control provisioned in a system requires a person to present his/her real identity that he/she intends to hide from privacy protection mechanisms. The literature review conducted by the author in Chapter 3 indicates that user-centric privacy protection does not appear to have been emphasized to date. Demand for flexible mechanisms to support active participation and choice of individuals to manage their privacy towards dynamic context-aware environments is expected to be

significant.

The context-aware paradigm, other than ubiquitous computing in general, is another facet of the author's privacy protection focus. The notion of ubiquitous computing includes a wide system and application scenario. It involves many different understandings that are driven from various research interests and purposes, including but not limited to sensor computing, location-aware computing, user-adaptive systems, etc. Context-awareness is one ubiquitous computing paradigm, and a key driver in ubiquitous computing [1]. It stresses the system's ability to discover and take advantage of contextual information (such as user location, time of day, nearby people and devices, and user activity) to make decisions about how to dynamically provide services or adapt to meet user requirements. This identification of context-aware computing as separate but part of ubiquitous computing does matter to this work. For one thing, the use of personal information to provision personalized services and applications is the key idea of context-aware computing, which asks for more stringent protection for individual privacy than other ubiquitous computing environments. For another, there are methods of providing access control and employing anonymity in ubiquitous computing systems, however, very little privacy work has been specially designed with context-awareness in mind, or which is capable of tackling privacy challenges resulting from context-awareness. The context-aware paradigm imposes some unique yet critical privacy needs (such as plausible deniability[2] and ambiguous disclosure[3]) that have not been adequately researched by the privacy research in ubiquitous computing.

For these reasons this thesis does not deal with the generalities of ubiquitous computing,

---

[2] Plausible deniability is a situation that potential observers of information disclosure cannot determine whether a lack of disclosure was intentional. According to [18], mobile phone serves as a good example, in that if a person does not answer a call, it could be a technical reason (e.g. being outside of reach) or for social purposes (e.g. not wanting to talk to the caller right now).

[3] Disclosing ambiguous information is desired, as quite often users' privacy preferences are not black-and-white but rather involve different levels of accuracy or inaccuracy. According to [17], ambiguous disclosure is often applied in two ways, either abstracting away some details of information, or providing false information on purpose, such as using pseudonyms to hide the real identities.

but focuses on privacy issues in the context-aware paradigm.

1.3 Major Contributions and Novelty Aspects

This thesis presents in detail novel approaches and key technologies that comprise the author's privacy solution. Major novelty aspects and contributions that help justify the author's effort among similar attempts are highlighted as follow:

First of all, the author has conducted a thorough survey and analysis of privacy work in the field of ubiquitous computing and some related fields, which contribute to a better (if not a full) state-of-art understanding of issues involved in protecting individual privacy in dynamic context-aware environments, its requirements and challenges.

Secondly, to the best of the author's knowledge, the author has been the first to introduce in the field of context-aware research a distributed privacy protection model. The model addresses the challenge arising from the dispersed nature of context information. It separates the privacy decision process from the enforcement process and thus allows multiple parties to contribute to information disclosure control with clear accountability. Behind the proposal of the distributed model is a new understanding of ubiquitous computing systems as a composite environment that takes into account heterogeneous types of ubiquitous systems and a discussion of the different parties involved in preserving privacy in such an environment.

Thirdly, the author has proposed an intelligent agent technology to address two key concerns of preserving privacy in context-aware ubiquitous computing environments: privacy feedback (i.e. notifying people of relevant information disclosure) and privacy management (i.e. allowing people to express their privacy preferences and manage privacy levels). The Privacy Agent provides mechanisms to support ambiguous

disclosure and some levels of plausible deniability, two key privacy requirements that have not been adequately addressed by many existing solutions. In addition, an important approach has been taken by the author to develop the Privacy Agent technology by integrating social and legal privacy mechanisms in the technical solution. This does not appear to have been emphasized to date by existing solutions. Also, unlike many existing solutions, the development of the Privacy Agent technology is independent of specific context-aware architectures, systems and middleware solutions. A prototype implementation using Web Service technologies (in Chapter 7) demonstrates that the solution is easy to deploy to achieve interoperability across different system platforms and devices, and can be scalable to the global Internet.

Fourthly, the author has developed a new lightweight Privacy Policy/Preference Language by adapting the World Wide Web Consortium (W3C)'s Platform for Privacy Preferences Project (P3P) practice [95], and shown it can be used to facilitate individual privacy expression and support automated processes of the Privacy Agent. Compared with few other similar research efforts that apply the P3P practice in the ubiquitous computing environment, the author's work provides a sound solution to construct privacy preferences to limit information disclosure not only with respect to data collecting policies, but also in response to dynamic contexts.

Fifthly, the author has successfully exploited ontology-based methods to represent the Privacy Policy/Preference Language, and developed novel hybrid reasoning mechanisms that combine ontology-based description logic and rule-based application-specific logic for semantic policy analysis of individual privacy preferences and context reasoning. Although using ontology for context information modeling has been attempted by other research groups (e.g. [46, 47, 48]), the ontology that focuses on expressing privacy vocabulary and policy language is entirely novel. Also, to the best of the author's knowledge, the semantic reasoning approaches taken by the author to perform privacy policy evaluation as well as to detect preference conflict and redundancy have not been presented by previous work. As a key contribution of this

doctoral research, the author's experience with ontology-based methods has revealed the potential of taking advantage of semantically-rich policy representation and reasoning to reduce human error, simplify policy analysis, detect policy conflicts, and facilitate policy understanding.

Last but not least, the author demonstrates through experimental evidence the feasibility of coupling rule-based methods with the semantic web programming framework (i.e. Jena [141]) to support semantic analysis of privacy preferences, in addition to context reasoning. The experimental work appears to be among very few successful attempts to find methods to validate the ontology-based approaches, due to the difficulty caused by both immaturity of ontology standards and a lack of tool support for reasoning axiomatic rules over ontology description. The experimental evidence shows that the author's proposal of using hybrid reasoning mechanisms can perform the task of semantic privacy policy evaluation, preference conflict and redundancy detection, and context perception properly, and lead to robust performance. The evaluation results contribute to privacy research in the field of context-awareness, by introducing novel semantic reasoning mechanisms in privacy preference and policy analysis, and by providing insights to address key performance doubts about using ontology-based methods in resource-constrained environments.

1.4 Thesis Structure

This thesis continues with Chapter 2 and Chapter 3 where background and related work on the context-awareness and privacy protection are presented and discussed. Chapter 2 ("*Context-Awareness Research*") introduces the important field of context and context-awareness. Chapter 3 ("*Privacy and Privacy Protection in Ubiquitous Computing Environments*") provides an understanding of the issues involved in protecting individual privacy in the dynamic context-aware environment, its

requirements and possible approaches. Significant work in the background study goes to literature reviews on prior and current research projects, and to survey and analyze various research approaches that they adopted.

Chapter 4 ("*A Distributed Privacy Protection Model and An Intelligent Agent Technology for Privacy Protection*") presents the author's proposal of a distributed privacy protection model for context-aware ubiquitous computing environments and work on an intelligent agent technology to overcome the limitations of existing privacy solutions and technical challenges that are identified in Chapter 3. The component design of the Privacy Agent technology and reasoning behind the design will be discussed. The author also discusses the limitations of the proposed solution, mainly in terms of security threats and trust concerns, and suggests some possible safeguard and mitigation measures.

Chapter 5 ("*Developing A Privacy Policy/Preference Language*") describes the author's work on developing semantic and syntactic specification of the Privacy Policy/Preference Language by adapting the W3C's P3P practice. It gives examples demonstrating how the language can be used by individuals to express their privacy preferences and by data collectors to state their data collecting policies. Reasoning behind major adaptation considerations will also be given. In the last section of this chapter, the author investigates previous ubiquitous computing work that attempted to apply the P3P practices in ubiquitous computing environment as well as existing efforts to develop policy languages for access control and security policies.

Chapter 6 ("*Ontology-based Modeling and Reasoning of the Privacy Policy/Preference Language and Context Information*") presents the author's work on exploiting ontology-based technologies to model the Privacy Policy/Preference Language developed in Chapter 5 and to use hybrid reasoning mechanisms for semantic policy analysis and context reasoning. The approaches taken by the author and reasons behind will be discussed. The final part of this chapter looks into emerging approaches that

employ Semantic Web technology and ontology-based techniques for policy representation and reasoning, in addition to related work on context modeling and reasoning.

Chapter 7 ("*A Prototype Implementation and Quantitative Performance Evaluation of Semantic Reasoning*") presents a proof-of-concept implementation of some key parts of the distributed privacy protection model and privacy agent technology proposed by the author. It helps shed light on how the privacy solution could be employed in a real world situation by simulating system environments in which it may be deployed. Reasoning behind implementation considerations as well as difficulties faced will be discussed. The author also conducts in the second part of this Chapter a quantitative performance evaluation on the semantic reasoning approaches developed in Chapter 6. The evaluation work helps validate the author's proposal of using hybrid reasoning mechanisms to perform semantic policy analysis of privacy preferences and context perception.

Chapter 8 ("*Conclusion and Future Work*") summarizes major contributions and novelty aspects of the author's work, and discusses some future work.

Figure 1.1 below illustrates the organization of key topics in this thesis, mapping their location and links with related sections.

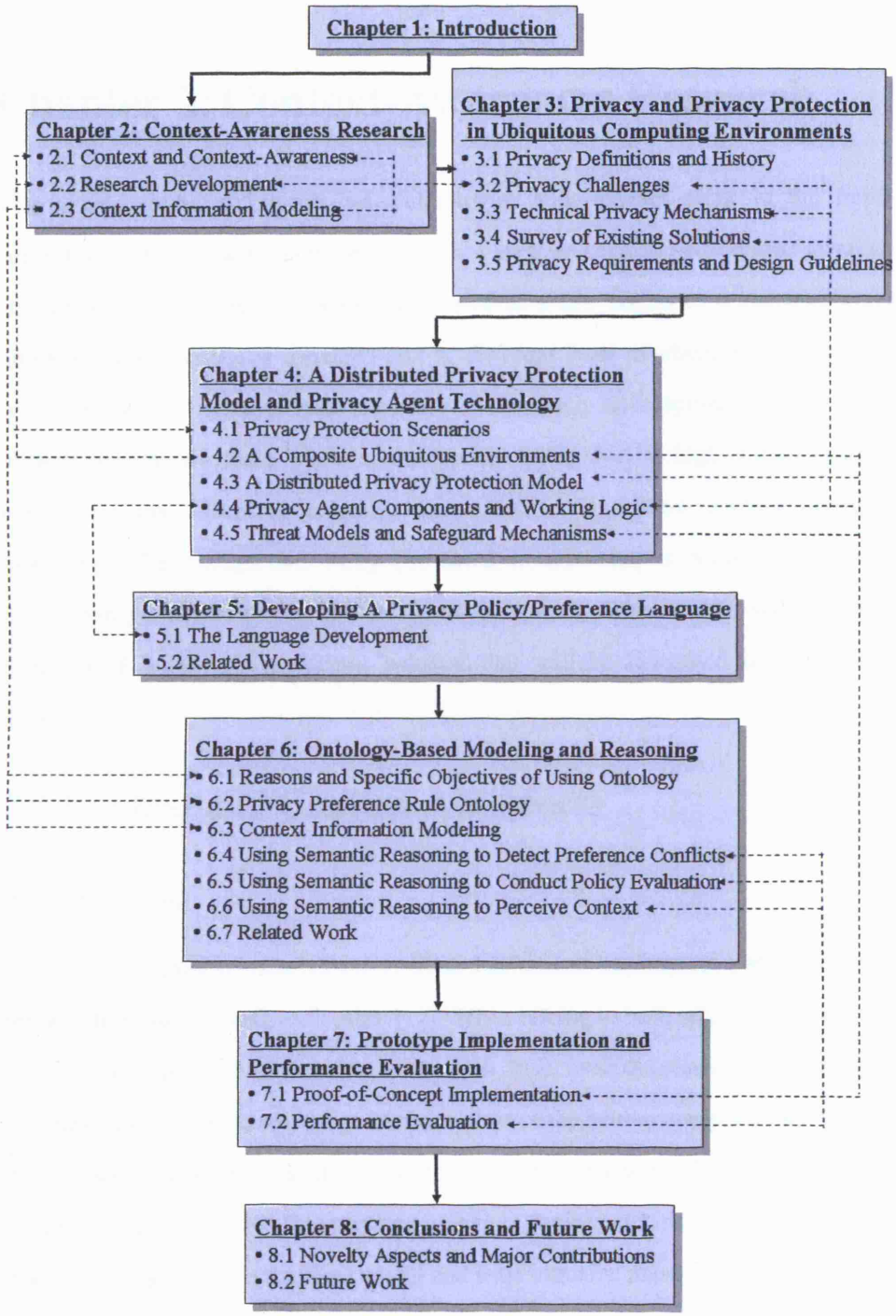

Figure 1.1 *Organization of this thesis, showing the location of key topics and their links with related sections*

Chapter 2: Context-Awareness Research

This chapter presents background technologies and relevant work in the field of context-awareness research. It begins with a look at context and context-awareness. Definitions are given of the two terms as well as a key notion to understand context information as composite concepts and at different level of abstraction. The chapter continues with a look at and summary of research development in the field of context-awareness to date. Some of the representative work is highlighted. Significant work after that discusses existing approaches to implement context information modelling. The background study presented in this chapter serves as a basis to understand privacy issues in context-aware computing environments and the author's proposal of a privacy protection solution that will be discussed in the following chapters.

2.1 Context and Context-Awareness

2.1.1 Context

The origin of "context" comes from an ideal that humans can interact with computers like the way humans talk with each other [12]. When talking to each other, people are able to use implicit situational information, i.e. context, to increase the conversational bandwidth. If computers could take advantage of the context of the human-computer dialogue, many new types of applications and services could be available. Those applications, for example, could help users find nearby services or devices, decide the best devices to use, and receive messages in the most useful and least intrusive manner.

The earliest research in the field of context-awareness attempted to define context either by examples or by synonyms. For example, the work by Schilit and Theimer [20] first introduces the term "context-aware", they refer to context as "*location, identity of nearby people and objects, and changes to those objects*", while some other

contemporary researchers such as Brown [21] refer to context as the *environment or situation.* Such concrete and rather straightforward definitions of context were considered hard to apply by Anind K. Dey, the pioneer researcher in the context-aware computing area. He provided a more general definition, i.e. context is *"any information that can be used to characterize the situation of an entity, an entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"*, and enumerated four primary contexts, including location, identity, activity and time that act as indices for other sources of contextual information [22].

The Dey's definition makes it easier for context-aware application developers to enumerate the context for a given application scenario. Many researches thereafter followed Dey to understand context in a general and loose manner, and tailor the Dey's definition to their own use. Work on network-centric context-aware applications by Yang et al.[23], for instance, adapts Dey's definition of context to be *"any information, obtained either explicitly or implicitly, that can be used to characterize one certain aspect of an entity that is involved in a specific application or network service. An entity can be a physical object such as a person, a place, a router, a physical link, or a virtual object such as IPsec tunnel, SNMP agent"*. The point is that if a piece of information can be used to characterize the situation of a participant in an interaction, then that information can be context only if it is relevant to the application or service that the entity is involved in and can be used to enhance it. In the example above, the capabilities of network nodes, such as bandwidth, throughput and operating system, are definitely context, but its date of purchase and years of service are certainly not.

This thesis follows Dey's and Yang et al.'s approaches to scope context information that is relevant to this work. Since personal privacy is the focus, *"any information, obtained either explicitly or implicitly, that can be used to characterize privacy aspects of an entity"* is more relevant to our work than the rest. The entity could be, as defined by Yang et al., either a physical object or a virtual object, but more often

entities are persons and devices that they carry and have the potential for divulging personal information.

Besides the general definition providing a comprehensive understanding of context, this thesis also advocates a notion of understanding context information as a composite concept and at different levels of abstraction. Low-level context abstraction includes raw contextual information often outputted directly from data sources, such as location coordinates provided by sensors and network throughput (in bps) rendered by network monitors. On the other hand, high-level abstraction is about social context, i.e. an abstract description of the real-world situation, such as "in a meeting". Context information of high-level abstractions often derives from information of low-level abstractions. The complexity of the process transforming raw context data to an appropriate level of abstraction, that is desired by applications or users, depends not only on the complexity of real-world situation that will be represented, but also on the architectural design of context-awareness systems.

Perceiving context as a composite concept helps understand the challenge of the distributed existence of context information in preserving individual privacy in context-aware environments. Figure 2.1 below exemplifies the information collected by a location-based context-aware application that notifies people when their friends and colleagues are nearby. The figure illustrates that a person's location information could be generated and gathered from various sources, e.g. sensors in an indoor environment, a GPS-enabled system that cover the outdoor range, and a calendar locator that records a person's appointments. Since the information in the context-aware paradigm is heterogeneous and usually distributed in the environment, it is less likely to be under the control of a centralized mode. Unfortunately, most of the existing privacy solutions did not tackle this challenge. They are based on the simplifying assumption that personal information is neatly partitioned into repositories that are under the control of a single user or system administrator in an enterprise environment. This makes them inadequate for preserving privacy in context-aware environments. The author will discuss further

the limitations of exiting approaches in the next Chapter.

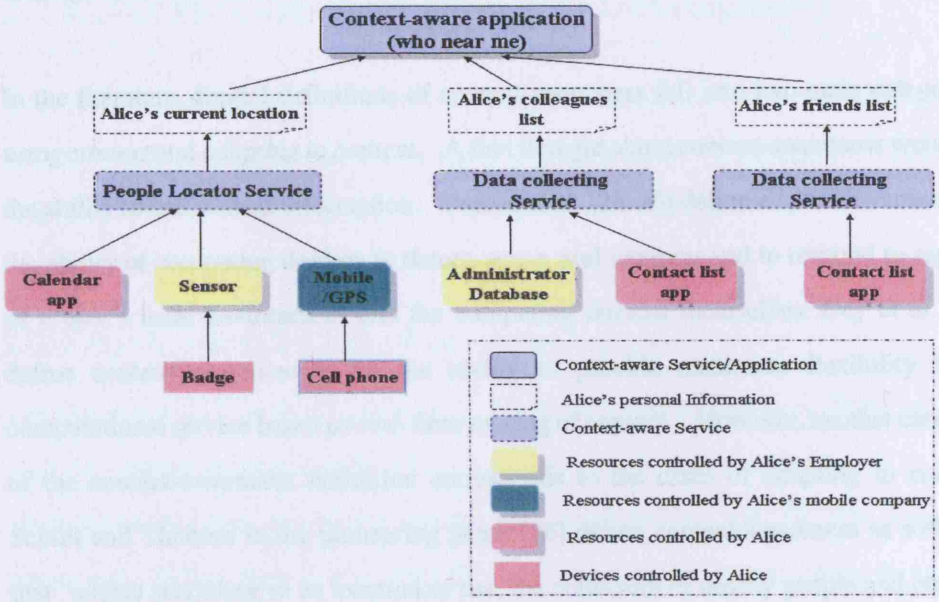

Figure 2.1 *The information collecting of a location-based context-aware application*

2.1.2 Context-awareness and Context-aware Computing

This section clarifies the concept of context-awareness, context-aware computing and context-aware systems.

The context-awareness concept has been increasingly gaining in importance in the area of distributed systems since the late 90's, as it seems a promising solution to a lot of problems that have been implied by the usage of mobile terminals in ever-changing environments [1]. A system's ability to make decisions about how to dynamically provide services or adapt to meet user requirements, based on and in response to, various contexts has become a key driver in the ubiquitous computing paradigm. This thesis refers to the use of context and context-awareness in ubiquitous computing as context-aware computing. In other words, context-aware computing is one ubiquitous computing paradigm which emphasizes taking advantages of contextual information to

make networks more receptive to users needs and enhance the users experience by making the communication easier and richer.

In the literature, formal definitions of context-awareness fall into two main categories: *using context* and *adapting to context.* A first thought about context-awareness would be the ability to use context information. Pascoe et al. [24, 25] define context-awareness as the ability of computing devices to detect, sense, and interpret and to respond to aspects of a user's local environment and the computing devices themselves. Dey et al. [26] define context-awareness to be the ability to provide maximum flexibility for a computational service based on real-time sensing of context. However, another category of the context-awareness definition corresponds to the cases of *adapting to context.* Schilit and Theimer in the pioneering paper [20] define context-awareness as software that "adapts according to its location of use, the collection of nearby people and objects, as well as changes to those objects over time." Brown et al. [27] define context-aware applications as applications that monitor changes in the environment and adapt their operation according to predefined or user-defined guidelines. These two definitions require that an application modifies its behaviour for it to be considered context-aware.

This thesis follows the definition postulated by Dey in [22] to describe context-awareness.

A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task. The user here can be an end human user or another application or entity.

This definition differs from the previous definitions by looking from the system's viewpoint to envisage and define context-awareness. Apparently, it is a more generic definition covering the applications that either use context, or adapt themselves to context. Furthermore, the idea of the user is extended from the human users to any entity and application that employ the context-awareness provisioned by the system.

2.2 Context-Awareness Research Development

Many researchers have so far studied context-aware computing and built context-aware applications to demonstrate the versatility of context awareness. Research development in the field of context-awareness has so far witnessed two trends.

2.2.1 Application domain of context-awareness

The first trend deals with the application domain of context-awareness. Early research work on context-awareness mainly focused on the use of location information to allow applications to adapt to a user's environment (examples are [28, 29 30]). Later on, researchers shifted their focus towards developing context-aware applications that exploit both user and environment contextual information to provide specific services (e.g., adaptive maps [31] and guided tours [32]) to users as they move. These approaches were restricted to the use of context information that lies within the user's immediate scope and ignored information related to the underlying network. More recently, for example, researches from EU's Ambient Networks project [33] and Context project [34] have recognized that network-centric contextual knowledge can play a major role in providing users with guaranteed services that can best utilize underlying networks. For instance, a streaming service for a mobile user can be optimized by making use of context information such as the capabilities of available end-devices and the network connectivity of users. As such, the concept of context-awareness has been broadened to include not only the characteristics about human users and their surrounding physical environment but also to include network related context such as connectivity, reachability, availability of resources, etc.

2.2.2 Software support for building context-awareness

Another trend in the field of context-awareness research is related to developing

software support for building context-aware systems and applications. Early systems were relatively simple, and were often constructed simply as distributed application components communicating directly with local or remote sensors. Active Badge [28], ParcTabs [31] and Cyberguide [32] are examples following this approach. The development of these applications and systems shows how computers can leverage even elementary notions of location, identity and activity to great effect, and demonstrated usefulness of context-awareness. However, it is also evident that the monolithic system design for specific applications is extremely difficult to build and maintain, primarily due to the diverse nature of context and devices that capture context [35]. Consequently, there is a need to separate the process of context acquisition from sensor operation.

The context-awareness research then has been moving on to provide architecture support for developing context-aware applications. The goal of the architecture support is to separate application development from system operations by providing a common framework that provides context information and operands (i.e. means to process context information on behalf of requesting entity) to context clients (entities, applications or services requiring context information to function). Framework and Toolkits (e.g. [24, 36, 37, 38]) have been devised to enable the creation of context-aware applications and services. While frameworks concentrate more on design reuse by providing a basic structure for a certain class of applications, toolkits build on frameworks by also offering a large number of reusable components for common functionality. Schilit's work [36] on ubiquitous computing in 1995 is probably the earliest attempt in this direction. In this Ph.D. work he proposed a general architecture trying to separate the application development from direct communication with sensors. Schilit's work gets credit for being first to articulate some of the difficult problems to be faced when building context-aware applications and by putting context-aware applications into practices. However, perhaps the most influential work providing such framework and toolkit support is Context Toolkit developed by Dey et al. [38] at Georgia Institute of Technology in 2001. The Context Toolkit is carried out as seminal

work on context-awareness by being first to produce a reusable framework for developing context-aware applications for all types of context, compared with Schilit's work [36] concentrating only on person and place context.

Building upon Schilit and Dey's experiences, quite a few research efforts thereafter have been carried out to provide architecture support to develop context-aware systems and applications. One of the appealing and influential researches is Context Fabric (Confab) proposed by Hong [39] at U.C. Berkeley. The Confab proposal advocates a "service infrastructure" concept, attempting to push as many system functionalities such as context collecting and processing into the infrastructure as network services that can be accessed by any device and any application. This is philosophically different from having the framework and toolkit support aforementioned, in that the service infrastructure can be used independently of hardware platform, operating system and programming language, thus achieving greater scalability [39]. In contrast, the framework and toolkit (such as Schilit's architecture [36]) are hardware dependent, their codes are stored and run locally on individual devices, every copy of the codes has to be updated whenever changes are made.

Both the Context Toolkit by Dey et al. [38] and the service infrastructure by Hong [39] are indeed middleware technologies. In both solutions, a set of additional system components were developed to promote reuse, improve maintainability, and reduce the complexity of deploying context-aware applications. The subsequent development of context-aware ubiquitous computing systems (such as Gaia [40], Solar [41, 42], PACE [43, 44]) has been strived to tackle basic issues traditionally addressed by middleware for distributed systems, including paradigms for coordination and communication between distributed components, in addition to developing architectural support for gathering and managing context information. While these ubiquitous systems have constructed middleware architecture with their own purposes and means, which are neither same nor exhaustive, Zhang and Todd [19] have proposed a shared conceptual model of four key functional requirements for developing generic context-aware

systems. The four functionalities are context collecting, context storage, context processing, and context dissemination.

The complexity of developing context-aware systems and applications makes middleware an essential requirement [19]. However, there remain quite a few critical technical challenges that must be overcome before even robust context-aware systems can be widely deployed and realistically evaluated. Preserving individual privacy is one of such challenges. Most of the existing context-aware systems provide very little support for privacy, although researchers often note the importance of privacy protection in the context-aware computing [12, 13, 14]. This drives the author's research to investigate potential approaches and mechanisms to offer adequate privacy protection for individuals to benefit from the advancement of context-aware intelligence.

2.3 Context Information Modelling

A paramount objective of the author's work is to develop a privacy protection solution that could protect various types of context information. Since the information in context-aware ubiquitous computing environments is heterogeneous, composite, and with various formats, this section presents a relevant discussion of background technologies that are used to model context information.

Early efforts to develop context-aware systems focused primarily on reusable components for combining and interpreting sensor outputs to derive high-level context information [36, 38]. With the advance of context aware computing, the importance of appropriate abstractions for gathering and reasoning about context information, have been recognized. Efforts have turned to developing formal models of context that integrate information from a variety of sources, support interoperation of context-aware

applications, and allow reasoning about context. The context modeling technologies help reduce the complexity of managing diverse context information by organizing, representing and describing context information in a uniform format and structure.

An informative work conducted by Strang and Linnhoff-Popien [1] presented a taxonomy of context modeling approaches. It classified various modeling approaches, based on the data structures that are used to express and exchange contexts, into key-value models, markup-based models, graphical models, object-oriented models, logic-based models and ontology-based models. A more recent work [45] by Balakrishnan et al. augmented this list by further introducing hybrid models that combine two or more of the listed approaches. Appendix A presents the author's understanding of these context modelling techniques and examines briefly their merits and shortcomings. The following part focuses on introducing and discussing the ontology-based modeling approach, as it will be employed in the author's privacy protection solution that will be presented, in particular, Chapter 6.

2.3.1 Ontology-based models

Increasing interest in ontologies in the last couple of years has led to emerging ontology-based context modeling approaches. Ontology-based context models have been independently developed by several research groups, including Chen et al. [46], Gandon and Sadeh [17], the Context ontology language (CoOL) [47] by Strang et al., and the Context Ontology (CONON) [48] by Wang et al.

The term "ontology" has a long history in philosophy, in which it refers to the science of describing the kinds of entities in the world and how they are related [49]. In the context of knowledge management, an ontology is a formal explicit description of the concepts in a domain, the properties of each concept, and restrictions on those properties [50]. The formality inherent in ontology-based models is particular useful in heterogeneous ubiquitous computing environments. For one thing, it promotes

information sharing and reuse. By describing contextual facts and their interrelationship in a precise and unambiguous manner, all participating parties in the heterogeneous computing environments can share the same interpretation of the information exchanged. By reusing existing ontologies that may be defined for different domains and with various purposes, efforts to compose a new ontology can be attempted without starting from scratch, and can benefit from model interoperability. For another, the formality gives rise to a certain level of reasoning capability and extensibility. Efficient reasoning is needed to derive higher-level context information from lower-level pieces of information. The emergence of representation tools and reasoning mechanisms (such as OWL [51] and RuleML [52]) allows the context model exploiting ontologies to be more expressive and powerful. Consequently, context-awareness research could have improved solutions to evaluate context queries, to check and solve inconsistent context knowledge due to imperfect sensing technologies, and to support the interoperation of different models.

The capabilities presented by ontology-based models to address critical issues including formal context representation, information sharing and logic-based context reasoning, however, could not or could not adequately be supported by other context modeling approaches. Figure 2.2 shows a graph indicating how different context modeling approaches combine support for the extensibility and reasoning capability. The X axis compares each approach's support for extensibility while the Y axis indicates the power of reasoning capability. Since the literature does not appear to provide quantitative evaluations of the feasibility of context reasoning in ubiquitous computing environments that always have to face resource-constrained devices, this map is not meant to be an accurate description but serves to give the reader a rough comparison of various modeling approaches. As illustrated in Figure 2.1, less formal context model such as Key-Value pair, Graphical Notation, Markup schema, and Object-oriental approaches are often based on proprietary representation schemes, and have fewer facilities to ease shared understanding about context between different systems. On the other hand, ontology-based and logic-based models have a stronger reasoning capability

than the rest, due to a high level of formality that they appear to have, and the fact that in the literature several reasoning and query engines have been proposed and are under development for the ontology-based and logic-based models, such as RuleML [52]. The high level of formality also gives the ontology-based model a great extensibility.

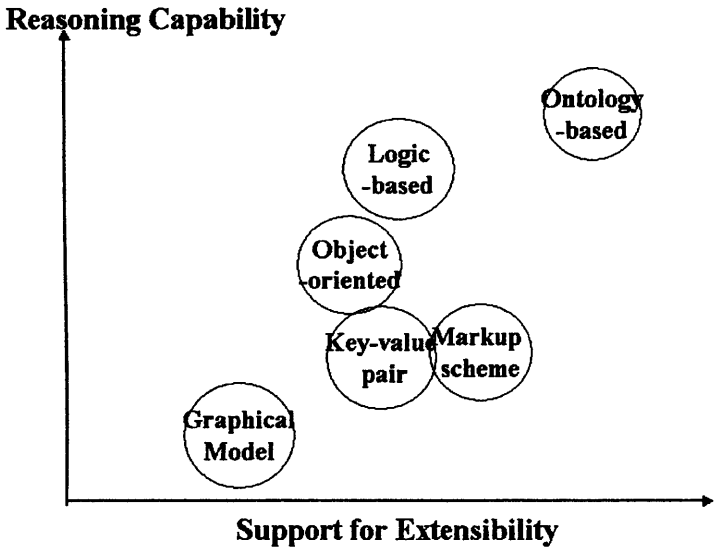

Figure 2.2 *Graph comparing the different context modeling approaches according to support for extensibility and reasoning capability*

Ontology-based models exhibit strengths in terms of a high degree of formality, semantic expressiveness and powerful reasoning capabilities. The strengths, however, do not come without a cost. In particular, the powerful yet complex reasoning mechanisms impose severe performance requirements on systems and computing devices that rely on ontological modeling technologies. As argued in [48], these authors indicate that ontology-based reasoning is not suitable for time-critical applications. Besides, creating or extending context ontologies is considered complex and error-prone, as the ontology languages are often verbose and unintuitive.

One way to address these problems is to combine the ontology-based modelling approaches with other, more natural solutions, such as graphical context modelling approaches. For instance, Henricksen et al. [53] investigated the incorporation of

ontology-based reasoning into their graphical modeling approach. This work showed an improved performance in terms of model checking and reasoning about model interoperability. Work on hybrid models, however, is still at an early stage. There are not enough details to allow a complete evaluation of such approaches. Nevertheless, some researchers certainly find the hybrid approaches promising. They argue that the hybrid approaches have "the potential of combining efficient information management with powerful reasoning mechanisms."[45].

The author's work on the generic privacy protection solution will employ the ontology-based method to model privacy-related vocabulary, including various types of context information. Performance evaluation work conducted by the author in Chapter 7 will present some insights and experimental evidence to address the performance doubt about using ontology-based methods in resource-constrained environments.

2.4 Chapter Summary

This chapter has briefly introduced the field of context-aware computing research. It provided readers with background information to enable a good understanding of the application of context, as well as the development of context-awareness. An important notion advocated by the author is to understand context information as a composite concept and at different levels of abstract. The author also proposed that the ontology-based modelling technology is more appealing to ubiquitous computing environments than other existing approaches, as it provides compelling capability to address critical issues including formal information representation, knowledge sharing and logic-based context reasoning. The background study conducted in this Chapter provides a good basis to understand the author's work on a privacy protection solution that will be presented through Chapter 4 to Chapter 6.

Chapter 3: Privacy and Privacy Protection in Ubiquitous Computing Environments

This chapter provides an understanding of issues involved in protecting personal privacy in dynamic context-aware ubiquitous computing environments, its requirements, challenges and possible approaches. It sets out by introducing the definitions of privacy and their history, and exploring privacy challenges in ubiquitous computing environments through reviewing literature work. The chapter then looks into possible technology offers that are available to preserve individual privacy, and discusses their applicability in the context-aware paradigm. Important work discussed after that focuses on a survey of context-aware prototypes and ubiquitous computing systems that have been specifically designed with privacy protection in mind. The most salient solutions are presented, and their versatility in tackling research challenges and in meeting users' privacy needs are evaluated. The survey and evaluation work have indicated that only a small subset of the privacy needs and challenges have been moderately addressed, and demand for adequate privacy protection in the context-aware paradigm is significant. The chapter ends with developing privacy design guidelines by exploring individual privacy concerns in context-aware environments and by examining existing legal and regulatory principles. The guidelines direct the author's efforts to develop a privacy-respecting solution that will be presented through Chapter 4 to Chapter 6.

3.1 Privacy Definition and History

One of the earliest attempts to define privacy goes back to the 19th century. In 1890, US Attorney S. Warren and his partner L. Brandeis defined privacy in their seminal law review article "The Right to Privacy" [54] as "the right to be let alone". This definition of privacy is actually a reference to the technical progress in the field of photography at that time. In 1884, Kodak invented the modern photographic film.

Instead of having to use heavy glass plates in studios, anybody could take Kodak's "Snap Camera" out on the streets and take a snapshot of anybody else without their consent. This definition created the basis for privacy tort law in the U.S. legal system [55].

A more updated and widely accepted privacy definition comes from the 1960s by legal and policy scholar Alan F. Westin. He defined privacy in his groundbreaking book "Privacy and Freedom" [56] as "Privacy is the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others". Interestingly, the origin of this definition was also driven by a technology advance at the time. In the 1960s, the invention of automated data processing was discovered by governments as an effective means to catalog citizens. Many European nations passed various data protection laws in order to prevent any misuse of such centrally stored information [14].

The Westin's definition is often referred to as Information Privacy and is probably the most important subject of discussion in many environments that handle personal information. As pointed out by Langheinrich, being in control of one's personal data is only one facet of privacy, privacy also covers other aspects such as *Media privacy, Territorial privacy, Bodily privacy,* etc. [14]. However, while other aspects of privacy have by now been well established in most legal frameworks around the world, often directly defined as constitutional rights, it is the information privacy that creates most trouble today. It is this kind of privacy that is especially relevant to the area of context-aware computing. The author will use Westin's definition of privacy in this thesis.

Looking back into the history, it is clear that the protection of personal privacy is by no means just a recent trend of the information society. The concept of privacy has been continuously shaped as new technology or social changes provided new ways of intrusion. Lately, the increased use of credit cards and the dawn of the Internet have

made privacy protection a hot-button topic. The notion of ubiquitous computing is probably another important technology that is altering society's conception of the reach and limits of privacy. A Privacy work in ubiquitous computing conducted by Hong proposed an interesting "Privacy hump" theory [57], in which he observed that the acceptance of many potentially intrusive technologies follows a curve that he calls "the privacy hump". Although this hypothesis is still a speculation at this point, it may help understand how to design and deploy ubiquitous technologies better and increase the likelihood that the technologies are accepted.

3.2 Privacy Challenges in Ubiquitous Computing Environments

The privacy tensions with ubiquitous computing was raised almost immediately when Weiser and researchers in Xerox Parc built an Active Badge system to first propose the concept of ubiquitous computing in 1991. Many people at Xerox Parc had visceral and highly emotional responses to their research. One researcher said:

> "Do I wear badges? No way. I am completely against wearing badges. I don't want management to know where I am. No. I think the people who made them should be taken out and shot... it is stupid to think that they should research badges because it is technologically interesting. They (badges) will be used to track me around. They will be used to track me around in my private life. They make me furious." [4]

However, it was not until 2001 that privacy concerns had gradually become a staple in ubiquitous computing conferences, leading to various mitigation mechanisms and technical solutions. In the field of context-aware computing research, in particular, much work has been done to develop context-aware middlewares, however, few of them

provided features for managing privacy although researches often noted the importance of privacy and security in context-aware computing [12,13,14]. This section summarizes, from reviews of the literature, three major reasons to help understand the difficulties involved in preserving privacy in context-aware ubiquitous computing systems.

First of all, privacy is a very personal issue and a malleable concept in practice. In addition to the general definition of information privacy, Alan F. Westin has even asserted that "no definition of privacy is possible, because privacy issues are fundamentally matters of values, interests and power" [58]. This reflects the variety of individual perceptions of privacy.

Different social and legal environments may lead to different levels of expectation from privacy protection. A survey conducted by Ackerman et al. [59] found three general groups in the American population: privacy fundamentalists, privacy pragmatists, and privacy unconcerneds. It was identified that the majority of the U.S. population are *privacy pragmatists* who are concerned about using their data, but the concerns were often significantly reduced by the presence of privacy protection measures such as privacy laws or privacy policies on Web sites. Few are *privacy fundamentalists* who are extremely concerned about any use of their data and generally unwilling to provide their data, even when privacy protection measures are in place[4]. A study by IBM [185] found that Americans slightly placed more confidence in companies handling of their personal information than did people from Germany or the United Kingdom. A more recent privacy report by MORI [186] investigated public awareness of and attitudes towards the sharing of personal information by interviewing 2098 adults throughout Great Britain and Northern Ireland. It was shown that 60% of the public say they are very or fairly concerned about public services sharing their personal information, with 22% *very* concerned. Only 12% say they are not at all concerned.

[4] A survey done now may yield different results, given that governments have posed more stringent data collecting requirements on personal data after the U.S.911 attack.

Varying levels of individual concern over privacy and information control may also result from different perceptions of risk and benefit. As pointed out by Hong [57], the adoption of credit cards serves as a good example. People are increasingly used to having to use credit cards for online shopping. This may be because they believe that the convenience of online purchases outweighs the potential cost of their transaction data being misused. A similar tradeoff exists in context-aware computing where people desiring to benefit from rich and interesting personalized context-aware services may be willing to compromise their privacy to some extent. Some privacy researchers (e.g. [115]) have attempted to introduce the risk and benefit evaluation in privacy disclosure decision. They suggested that benefit is a good incentive to let people disclose their information.

Individual privacy preferences towards dynamic context-aware environments could be more complex and more difficult to handle than those in traditional data management environment. As pointed out by Henricksen et al. [60], the environments typically contain collections of heterogeneous information with variable privacy requirements resulting from differences in users' individual privacy preferences, differences in the sensitivity of the information, and changes in users' privacy preferences over time and in response to context changes. This lack of common grounding regarding privacy requirements may make it difficult to have reasoned debates on what the potential risks are, and what potential solutions can be applied to address those risks.

Secondly, privacy is not a purely technical issue; preserving privacy also involves social norms such as ethics and trust, legal frameworks, and regulatory provision. Early implementers of user privacy tools, Hong and Landay [61], have noted that there is no perfect privacy technology, as there are many social and organizational issues that simply cannot be managed by technical means alone. Another implementer, Langheinrich [55] listed public safety and personal security reasons as important driving factors for less, rather than more, privacy. He argued that just as in the conventional

computing systems, simply not storing identifiable information might run counter to security and safety concerns of both society in general and service providers in particular.

It was learnt, from the early implementation of privacy solutions, that technical tools indeed interplay with legal and social mechanisms when preserving individual privacy. On the one hand, social and legal mechanisms play a tangible role in protecting individual privacy. For example, while one might be able to construct technology that completely anonymizes two or more parties that electronically interact with each other, legal frameworks set forth to protect society from crimes would try to prevent us from fielding it. As argued by Kobsa and Schreck [62], the deployment of personalized anonymous interaction strongly hinges on social factors, such as regulatory provisions that mandate anonymity and pseudonymity in access to electronic services. On the other hand, as Lessig noted [63], the way a technology is designed has significant impact on how other forces, including laws, market forces and social norms, can be brought to bear on the problem. At this point, Ackerman et al. [64] suggested that "we will need to understand, as a research problem, how regulatory and technical solutions might be co-designed to form a public good, and where are the boundaries between technology, law, and social norms in terms of individual privacy protection". The boundaries, however, are hard to build, as privacy is not an absolute interplay that must constantly be reexamined, depending upon technical possibilities and social needs. This difficulty led many researchers of ubiquitous computing systems to simply leave privacy as future work, as it was a problem that could never be "solved" in the technology sense.

Thirdly, context-aware computing poses new technical challenges that have not typically been a focus of privacy research in conventional data management environments, and also have not been adequately tackled by current ubiquitous computing researchers. Some of the major challenges are discussed as follows:

The first challenge is about information sharing. Context-aware computing often works

in social settings, e.g. to bridge distant family members or to provide awareness among a group of friends. An important idea is that in the context-aware paradigm people *do* want to share information with others. This increases the complexity of providing privacy protection, as individual's privacy concerns are not only over commercial or institutional services (e.g. a building's security system), but also over other participants involved in the same system or using the same applications. Also, this challenges conventional methods such as using anonymity and pseudonymity to hide real identities and limit user interactions; since parties that are receiving another person's information perhaps already knows his/her identity, and are not adversaries in any traditional sense. As argued by Hong and Landay [61], privacy risks involved in context-aware paradigm may be as simple as wanting to avoid undesired social obligations or potentially embarrassing situations.

The second challenge comes from the composite nature and distributed existence of context information. As discussed in section 2.1, context-aware systems rely on various types of context information in order to make decisions about how to dynamically adapt to meet user's requirements. This information is heterogeneous and usually derived from a range of sources, including user profiles, applications, sensors, etc. And in many cases, it becomes meaningful contextual knowledge desired by context-aware applications only after going through appropriate levels of aggregating, correlating and inferring raw context data from multiple sources. The composite nature of the information invalidates usual approaches to assign ownership and control of resources. Perhaps, privacy solutions have to take into account privacy protection along the way, i.e. how information is gathered, stored and processed [16], as there are perhaps no straightforward links between information sources (e.g. a sensor in a meeting room) and entities (e.g., a person captured by a sensor) that are entitled to control the corresponding information for privacy purposes. Furthermore, since information is distributed, it is probably not under the control of a person or single unit of administration in the system. This makes conventional access control mechanisms that are deployed at single points, like a file in a file system or an object in a database, no

longer work [16].

The third challenge involves providing unobtrusive user participation. Since information in ubiquitous computing environments could be collected easily without the users' knowledge, facilitating active participation of individuals in controlling their personal information is a staple requirement to preserve individual privacy in such environments. Indeed, central to the privacy requirements of dynamic context-aware environments is the need to empower people to choose information disclosure with the right people and services, in the right situations, and at the right level of detail, in addition to the ability to choose not to have information collected. Yet, the task of taking full context-aware controls over how their personal information is shared can be overwhelming to users (due to sheer volume, especially at the sensor data level). The task might disrupt their ongoing activities, which defeats the basic goal of making context-aware environments unobtrusive [64].

A survey and analysis of salient research work in the field of context-aware ubiquitous computing conducted by the author has indicated that only a small subset of the privacy needs and challenges identified above have been moderately addressed, and that the demand for adequate privacy protection in context-aware paradigm is unlikely to be decreasing in the current environment. Section 3.4 will discuss in detail the limitations in approaches taken by existing solutions to date.

3.3 Technical Privacy Mechanisms

This section provides an overview of some well-known and relevant privacy technologies, mainly drawing support from the area of Internet privacy and computer security in general. The goal of the survey is not to give an exhaustive list, but instead to select from possible technology offers and to describe those that could be and are

applied to the context-aware paradigm. Various privacy technologies, mechanisms and tools are scoped into five categories: identity management through anonymity and pseudonymity, access control, encryption and security mechanisms, privacy meta data, and the notion of computational trust. The most salient aspects of the background technologies and relevant work are presented. Their merits as well as their applicability to the context-aware paradigm are discussed.

3.3.1 Identity Management Tools——Anonymity and Pseudonymity

As a first thought, privacy threats emerge as a result of the linkage between a person's identities and his private information. Quite a few privacy solutions in conventional data management environments and in ubiquitous computing systems have thus tended to focus on providing identity protection. The underlying assumption is that if personal information collected cannot be traced back to that person, the collecting and use of such information pose no threat to his or her privacy. Anonymity and pseudonymity are two major identity management tools.

Anonymity is defined as "the state of being not identifiable within a set of subjects." The larger is the set of subjects, the stronger is the anonymity [65]. Being fully anonymous may be ideal from the privacy protection point of view, but it prevents individuals from using applications that require authentication or offer some form of personalization. As an alternative, pseudonyms enable service personalization without having to disclose real identities of people. In essence, a person is allowed to choose a unique but otherwise uncontrolled pseudonym by which she can be repeatedly identified until she changes to a different ID. Using the same pseudonym more than once allows the holder to personalize a service or establish a reputation, while always offering her the possibility to step out of that role whenever she wishes. To make the pseudonym legitimate while keeping real identities hidden from communicating parties, a trusted third party is often required. Such a third party will assign certificates to pseudonyms. It

can also revoke a pseudonym if it is misused, or help a transaction party to trace back to real identities if necessary. Anonymization research such as [66] and [67] presented examples of employing pseudonymity mechanisms to hide a user's real identity. They combined encryption techniques, involving digital signatures, blind signatures, pseudonyms and trusted third parties, to help achieve anonymous yet authentic communication in an e-community environment and in a reputation system.

Privacy work such as [62, 68, 69, 70, 116] has presented how pseudonyms can be used in ubiquitous computing environments and has shown that different levels of pseudonymity and implementation exist. The Active Badge system [68] developed at Xerox Parc is one of the earliest location systems that addressed privacy issues through pseudonymity. It proposed that in a smart office environment each user wears a badge and has a User Agent to collect and control access to his/her location information. The badge uses a pseudonym, instead of real identity of its owner, to be identified by its owner's User Agent. Since any request for a user's location information must be routed through his/her User Agent where regular access control mechanisms can control dissemination of the user's real identity, the user could benefit services without disclosing his/her real identity. A more recent pseudonymity-based privacy solution was proposed by Roussaki et al. [116]. In the distributed context management system developed by them, a person's context information was partitioned into a set of smaller sets, each of which was assigned with a virtual identity (i.e. pseudonym) of the person. The partition of personal information and the application of multiple virtual identities help resemble a restricted view on a person presented to external parties, and to some extend prevent the linkage between a person and his real identity.

Compared with the fairly simple pseudonymity approaches in [68, 116], other more sophisticated pseudonymity solutions including [62, 69] have proposed stronger identity protection by building underlying anonymous communication infrastructures, so that anonymity can be achieved at the communication level. In particular, Kobsa and Schreck [62] presented an informative discussion about employing different levels of

anonymity in user-adaptive systems, and promoted the use of mix techniques to achieve communication-level anonymity. The *mix technique* was originally introduced by Chaum [71]. It provides sender anonymity as well as receiver anonymity through public key cryptography. The basic idea is that data are not sent directly to a destination address but instead repeatedly encrypted with the public key of multiple mix nodes. The mix nodes are publicly known computers that participate in the mix network. This in effect chains a number of decrypt-and-forward operations, and makes it difficult to analyze traffic. The disadvantage of this approach is that routing data across these mixes adds a fair amount of latency to network traffic, and only provides anonymity. A similar concept to mix networks is mix zones proposed by Bersford and Stajano [69]. They developed an anonymity service architecture that mixed pseudonyms of all users currently located in an area, and delayed and reordered messages from subscribers within a mix zone to confuse observers. The problem with this approach is that there must be enough subscribers in a mix zone to provide an acceptable level of anonymity [72].

Another interesting solution to location privacy was a k-anonymity-based approach proposed by Gruterser and Grunwald [70]. Unlike the previous methods, where anonmyonity was used to separate the linkage between real identity and personal information, k-anonymity explores the use of annonymization to obscure the location information disclosed. In essence, a trusted proxy is employed to hide a user's precise location by returning an area that has k-1 other people. Gorlach et al. [72] provided an informative discussion of various types of attacks on location privacy, and presented a survey of existing location privacy solutions.

Discussions

Given the difficulty of asserting explicit consent in electronic communications, anonymity and pseudonymity technologies get the credit for offering the possibility of easing the complexity of managing privacy requirements. Not only are they an

important option when providing users choices so that those who wish to remain anonymous can remain so, they also allow the legal collection of certain types of data without revealing real identity of individuals. Modern privacy legislation required that "anonymous or pseudonymous access ..... must be offered whenever technically possible" [73].

However, there are obstacles to apply anonymization approaches in the context-aware paradigm. First of all, the anonymity and pseundomity only address a relatively narrow aspect of privacy and do not provide a useful level of privacy when communicating with people in the context-aware paradigm where people *do* want to share information with others. For example, one could imagine sharing location information among friends; if the friends are requesting the person's current location, they probably already know the person's identity, and hence anonymity is not useful.

Secondly, better identity protection through anonymous communication techniques such as a mix network [62] and a mix zone [69] introduces communication complexity, which may add a fair amount of latency to network traffic and only provides anonymity. Also, building anonymity at the lower level of communications perhaps is a nontrivial task for many existing context-aware systems that are currently not based on anonymous communication architectures. Considering the mobile network as an example, devices that access the network must be related to subscriptions, and thus to users.

Thirdly, even the most sophisticated anonymity and pesudonymity solutions might not be able to prevent the linkage between personal privacy information and real identity, once some kind of additional information is correlated. A ubiquitous computing work by Sweeney [74] has shown that anonymously collected information can easily be combined later to form identifiable information. Similar work has been done by Beresford and Stajano [69], who analyzed the use of pseudonyms in location-based context-aware system. Using simple heuristics, such as a person's office and which

pseudonym spends the most time at this desk, the researchers were able to correctly identify all pseudonymized users in a location system deployed in their lab.

3.3.2 Access Control

In contrast to identity protection through anonymity and pseudonymity, systems relying on access control focus on controlling the circumstances of data release, e.g. selectively allowing or disallowing others to see or obtain a user's personal information, rather than hiding user identity. In ubiquitous computing, such systems typically build upon conventional access control mechanisms with some specific extension for the ubiquitous environment. For instance, some location-based context-aware applications [75, 76, 77] extend existing access control mechanisms, using location parameters, in order to allow access rules that apply to a particular geographic region.

Classic access-control models, such as Access Control Lists (ACL) or Role-Based Access Control (RBAC), provide no, or very limited, support for the context-aware paradigm where context-sensitive access control to confidential information is required. As argued by Zhang and Parashar [78], user's access privileges in ubiquitous computing environments will not only depend on "who the user is", but also on "where the user is" and "what is the user's state and the state of the user's environment". As a result, access control mechanisms that change the privilege of a user dynamically based on context information is required.

This limitation has been tackled by some existing research [78, 79, 80, 81, 82]. These researchers attempted to add context awareness to Role-Based Access Control (RBAC) by augmenting or replacing traditional user attributes such as identity with contextual information. In particular, Covington et al. [79] were among the first to use context to provide access control for a Smart Home environment. They proposed a Generalized Role Based Access Control (GRBAC) model that enhances traditional role-based access

control by introducing "environment roles". Environment roles can describe any state of the system, such as locations or times and can be used to implement controllable properties. Like GRBAC, the access control model proposed by Neumann and Strembeck [80] also made the assignment of permission to a role conditional on current context. The context concerned in its model includes, for example, time and user location. The problem with the GRBAC model, as criticized by Zhang and Parashar [78], is that a potentially large number of environment roles defined in GRBAC may make systems hard to maintain, which loses the advantage that RBAC provides. Instead, Zhang and Parashar [78] proposed a Dynamic Role Based Access Control (DRBAC) model that is able to dynamically grant and adapt permissions to users according to current context. Context information in Zhang and Parashar's work includes user environment (e.g. user location, time that a user accesses resources) and system information (e.g. CPU usage and network bandwidth). Huang et al. [81] presented another effort to extend role-based access control. They proposed a context, rule and role-based access control (CR-RBAC) model for enterprise ubiquitous computing environment. In this model, context-based access rules are defined to describe the constraint relation among user-role-permission assignments, which replace manual operations that assign users to roles or roles to permissions in RBAC. Another interesting work to enhance RBAC was developed by Guo et al. [82]. They proposed a trust-based access control, and attempted to incorporate trust negotiation mechanisms in performing role-based access control for ubiquitous computing applications. In their proposal, once initial trust is established, each user is arranged a role subset, and context information is used to decide which role is active.

Aside from the above efforts to extend the Role-based Access Control model, privacy work in the field of ubiquitous computing have looked into another important model, i.e. rule-based access control. The rule-based model has been developed by many salient ubiquitous computing systems (e.g. [17, 75, 76, 77, 84]), although with considerable variation in rules that systems wish to have governed context and privacy. For instance, Hengartner and Steenkiste [16, 77] specifically addressed location privacy as part of the

AURA project [83], allowing users to formulate location disclosure rules that operate not only on the identity of requestors but also on current time and location of the users. Additionally, rules can explicitly set the granularity level of return location information. Similarly, the Houdini system developed by Hull et al. [75] also provided a rule-based approach to adjust the disclosure of location data according to user-defined granularity (e.g., returning only the city name instead of the exact street address). In addition, Gandon and Sadeh [17], Chen et al. [46], and Al-Muhtadi et al. [84] each employed centralized rule engines for running access control.

Discussions

Compared to Role-Based Access Control (RBAC) and Access Control Lists (ACL) mechanisms, rule-based access control is able to offer a better fine-grained access control, making it more desirable for the ubiquitous context-aware computing environment. For one thing, in context-aware systems, flexible access control mechanisms are required not only for controlling the access, flow, and retention of personal information, but also for controlling the precision of information disclosure in order to achieve nuanced social ends, such as ambiguous disclosure and plausible deniability (as explained in Chapter 1 and section 3.5.1). These cannot be achieved without finely-defined rules with respect to various data subjects, purposes, recipients, conditions as well as differential sensitivity of information; the fairly simple mechanisms of ACL and RBAC —— by assigning permission either directly to low level data objects, or to specific roles and their associated operations —— are inefficient and apparently less applicable in these cases. For another, declarative policy languages (such as XACML[85]) employed by rule-based authorization to form various access control rules, are considered to be well matched to, compatible with, and desirable for context information that is characterized by semantic richness.

While the rule-based control offers increased flexibility, it also has some disadvantages. It may be too complex for individuals or administrators to manually set up rules or

decide on a case-by-case basis what level of granularity is most appropriate under certain conditions. It is proposed that since people can hardly predict all possible scenarios for granting or denying access to their information in response to dynamic context, flexible mechanisms must exist to not only facilitate users to articulate their privacy requirements for the first time, but also to give them context-aware controls over how their data is shared, i.e. enabling them to make any changes on their initial preferences whenever needed[5].

Regardless of the control models employed, there has been a major problem with ubiquitous computing systems that addressed privacy by provisioning access control mechanisms. These systems [17, 46, 75, 76, 84], either explicitly or implicitly, assumed that there would be a centralized administrator who configures the access control mechanisms. By accepting this simplifying assumption of the centralized control model, these solutions have not been capable of tackling the challenge provided by the composite nature and distributed existence of context information. One notable exception is the distributed access control work proposed by Hengartner and Steenkiste [16, 77]. They presented a salient work focusing on distributed access control mechanisms for location information in ubiquitous computing. With a clear analysis of the dynamic nature and intricacy of relationships between different pieces of context information, the access control mechanisms and architecture proposed by Hengartner and Steenkiste are considered here to be more realistic.

3.3.3 Secure Communication and Encryption Tools

Security measures provide, in part, the means to achieve privacy. Like in conventional data management environments, security requirements for ubiquitous computing environments comprise three key aspects, i.e. confidentiality, integrity and non-repudiation, but in a more stringent way.

[5] The design of Privacy Agent technology proposed in this work aims to incorporate these mechanisms, which will be presented in section 4.4 Privacy Agent Components and Working Logic.

Confidentiality

Confidentiality ensures that information is accessible only to those authorized to have access [87]. The disclosure of sensitive information would not be difficult in ubiquitous computing environments where information collecting and dissemination happen frequently, and sensitive information might live indefinitely and appear anywhere at anytime.

Integrity

In cryptography and information security in general, integrity refers to the validity of data, i.e. recipients should be able to determine if a message has been altered during transmission [87]. Integrity can be compromised in two main ways: either by malicious altering or by transmission errors. Integrity mechanisms mainly target protection against unauthorized modification of information.

Non-repudiation

Non-repudiation refers to preventing a party from denying previous commitments or actions [87]. Non-repudiation is not a core security requirement of many systems, however, in the ubiquitous computing systems, where multiple parties may be responsible for information and contributions to privacy decisions, non-repudiation mechanisms are thus useful. They can, for example, prevent service providers from denying data collecting, and context providers from denying context information provision.

Encryption mechanisms provide methods to protect information confidentiality and integrity by securing the communication between data subjects and data collectors. Encryption can be implemented in two main ways: either to control private information perception by using file or object encryption to obscure the data before transmission, or to use cryptographic protocols to perform network traffic encryption, such as SSL (Secure Sockets Layer) [90]. A privacy-aware solution (PawS) developed by

Langheinrich [55] employed SSL and digital signatures to secure interactions between parties involved in SmartSpace environments, while the security scheme developed in the AURA [83] and Gaia project [40] gave examples of the former. More specifically, in AURA, Hengartner and Steenkiste [91] proposed an encryption-based access-control scheme where services provide confidential information to any client, but only in an encrypted form; access rights are represented by a decryption key. In Gaia, AL-Muhtadi et al. [84] also introduced an encryption-based access control framework for location-aware systems, where information in the systems is stored in an encrypted fashion, and can be aggregated and decrypted only when a requestor's location has been verified.

Other staple cryptographic tools involving digital signature and blind signature[6] have been exploited by ubiquitous computing work to aid privacy protection mechanisms through anonymization and access control. For instance, the Geographic Location and Privacy (Geopriv) specification [76], proposed by IETF working group for location-based context-aware systems, employed digital signature to protect location information from illegal redistribution. An access control model developed by Ren and Lou [92] integrated two cryptographic primitives, blind signature and hash chain[7], to enhance the anonymous authentication process. The blind signature is also employed by anonymization researches such as [62, 67] to achieve anonymous yet authentic interactions.

Discussions

Security mechanisms and best practices provide some privacy capabilities for sensitive information distributed in ubiquitous computing environments, but they do not meet the privacy protection requirements for responsible and respectful acquisition and management of personal data. The security mechanisms are not sufficient to safeguard against subsequent use (once data is disclosed), to minimize the risk of sensor-based

[6] An introduction of Blind Signature can be found in [88].
[7] An introduction of Hash Chain can be found in [87].

disclosure or to reassure users.

Applying encryption and cryptographic tools in context-aware ubiquitous computing environments needs to take into account resource-constrained factors. The environments consist of various kinds of devices that offer different computational capabilities in terms of memory, CPU speed, battery lifetime, display size and support of input peripherals. An encryption algorithm that works for a new generation of PDA may not be expected to run equally smoothly on a low-power mobile phone. Hengartner and Steenkiste [77] built an example implementation of their access control model for people location information based on SPKI/SDSI certificates. They provided experimental evidence, showing that the delay introduced by setting up secure connections is significant.

3.3.4 Privacy Meta Data

Security tools as well as access control models typically do not deal with copy protection and distribution when information is disclosed. Solutions to address this issue have been investigated under the broad research area of digital rights management (DRM). There have been some research efforts attempting to borrow ideas from digital rights management to manage personal information in ubiquitous computing environments. The notion of Privacy Meta Data, means data about privacy data, is introduced by the author to describe the privacy declaration that governs information collecting practice and sequent use.

One of the first Privacy Meta Data solutions for ubiquitous systems was proposed by Jiang and Landay [93]. They introduced a notion of *Information Space* to organize information, resources, and services around privacy-relevant contextual factors (e.g. a group of owners who decide permissions), and developed a privacy tagging system that consisted of unified *Privacy Tags* attached to every part of information The *Privacy*

Tags describe which users can perform which operations within a certain information space. Hong and Landay [61] at U.C. Berkeley employed the idea of *Information Space* and *Privacy Tag* when developing a privacy-respecting context-aware architecture, called the Confab Toolkit. In Confab, *Information Space* was materialized as network-addressable logical storage units that store context information about a single entity, i.e. a person, a location, a device, or a service. *Privacy Tags* were implemented as a means for owners of information space to declare what they want data collectors to do with their information, independently of data collectors' plans. More specifically, *Privacy Tags* specify when owners' data should be deleted, to help enforce limited data retention; how long data should be retained before being deleted; the maximum number of previous values that should be retained (e.g. a value of 5 means only retain the last five places a person was at); an address to send notifications when owners' data are used by others. *Privacy Tags* are optionally attached to information and travel with it, governing data practice when the information leaves an Information Space.

Another important example of Privacy Meta Data is, the *Privacy Contract*, proposed by Langheinrich [55]. In developing a Privacy-aware Solution for ubiquitous computing environment (PawS), Langheinrich introduced the concept of a *Privacy Contract* to allow data collectors to state the required or preferred personal data, the data collection's scope, and the data collection's intended use and consequences. *Privacy Contracts* in PawS are similar to *Privacy Tags* in Confab, with respect to the idea of using metadata to enforce privacy compliant usage and retention, but they differ clearly in their use: while *Privacy Tags* unilaterally declare what data owners want data collectors to do with the data, independently of data collectors plans, *Privacy Contracts* in PawS are declaration by data collectors that data owners basically either rejects or accepts (potentially with a range of options). The difference demonstrates two alternative approaches to facilitate users to express their privacy preferences. Whereas the method taken by PawS eases the complexity of managing users' privacy preferences from a system operation perspective, the Confab approach, by letting users take the

initiative to articulate what and how they want to control, is apparently a more user-centric yet complex method. More interestingly, the PawS also applied the privacy meta data concept in database management. It developed a Private-aware Database (PawDB), by which not only the data itself but also privacy policies that describe the data's allowed usage, dissemination, and retention, are stored along with the data, so governing data access.

The PawS's approach to let data collectors declare policies for governing data collecting practice has also been adopted by two other privacy-respective ubiquitous computing systems by Myles et al. [15] and WASP by Zuidweg [94]. Like Langheinrich's proposal [55], the policies applied in these two systems were developed based on employing policy elements and terminology in the "Platform for Privacy Preferences Project" (P3P) specification [95] developed by the World Wide Web Consortium (W3C)[8]. A similar policy language to the P3P is the Enterprise Privacy Authorization Language (EPAL) [96] developed by IBM. It was designed for expressing access rights to information in enterprise environments. Like the P3P practice, the EPAL is used only as a vehicle for data collectors to declare data collecting practice, but not as an appropriate language to be manipulated by information owners to express their privacy requirements. Compared to the widely-used P3P practice, the application of the EPAL outside IBM appears limited.

Discussions

Using meta data to govern privacy compliant usage and retention is a good practice for context-aware environments where hiding identity is not desirable. It enhances users' trust in disclosing their personal information, by empowering them to declare privacy requirements, and by providing them with additional background information about the

[8] The P3P is an attempt to find privacy mechanisms for the Web. It was launched in May 1997 at the World Wide Web Consortium (W3C) in an effort to develop a specification and vocabulary to instruct websites to announce their privacy practices in a standardized machine-readable format that can be retrieved automatically and easily interpreted by users' browsers [95]. A brief introduction to the P3P and P3P-based method will be presented in Appendix B.

disclosure, its conditions, and third parties involved. This links directly into social mechanism of trust, as it can provide assurances about which users can make trust decisions with incomplete knowledge about data collectors and other parties involved.

As demonstrated by the Confab [61], PawS [55] and WASP [94], applying privacy meta data in existing context-aware architectures does not require major changes to the core functionality of the existing architecture. It could be implemented across platforms and at the application level when choosing an appropriate policy form such as the XML scheme. Major work happens at the client side, in which appropriate notification and user consent mechanisms are required.

A major concern with the privacy meta data solutions is that the approach counts on a trustful social environment and legal enforcement to function effectively, i.e. it relies on social and legal pressures to compel data collectors and third parties to comply with their stated policies or announced privacy policies by data subjects. In cases where there is a strong level of social trust, regulatory effort and market motivation, this will suffice and can help prevent accidental disclosures. In cases where there is not a great deal of trust, other mechanisms, such as anonymity and encryption tools, may be required. But unfortunately, existing solutions such as Confab [61] and WASP [94] do not have any built-in support for such remedial mechanisms.

3.3.5 Computational Trust

Privacy protection through privacy meta data relies on social trust, i.e. merely "hoping" that data collectors will adhere to posted privacy policies. In contrast, researchers in the field of network security have long used the term *computational trust* as an alternative to the social trust. The computational trust is originally a concept for decentralized access control systems, which computes whether a certain certificate holder is authorized for a specific transaction without relying on a central registry [117]. The

subsequent development has made the computational trust a rather fashionable topic in many fields of computing research. For instance, the mobile and autonomous agent community has used the trust to automate cooperation between agents. It typically entails reasoning about the agent's intent, competence, availability and promptness, rather than verifying a set of credentials [118].

In the field of ubiquitous computing, the computational trust has been employed to solve the problems of granting or denying access to certain resources (e.g. [119]), selecting services in dynamic environments (e.g. [120]), and exchanging data between entities (e.g. [121]). Many researches have expanded the trust concepts from network security; they grant or deny access not simply based on pre-computed certificates, but rather depending on a particular context. While some researches have simply tried to incorporate generic context variables into their solutions (e.g., [122,123]), others have explicitly based the computation on concepts from psychology, such as dispositional or situational trust and beliefs (e.g., [124]). Their idea is to take established trust concepts in social sciences and use them to build something similar to human trust into their solutions.

Discussion

This notion of decentralized control makes the computational trust an alternative privacy model in ubiquitous computing environments, since such environments often tend to use distributed system architectures. In addition, ubiquitous computing systems are expected to operate in a non-intrusive fashion. By allowing computing agents to compute the "trustworthiness" of an electronic counterpart based on past experiences and/or third party recommendations, it can free users from such banal things as usernames and passwords, and free the computing agents from the tasks of soliciting and comparing access policies in the environments that are characterized by intermittent disconnection and highly dynamic access patterns.

However, using the computational trust to replace the dependency on social trust is not without doubt. An important yet often overlooked aspect is the validation of the computational trust. Little work on trust in ubiquitous computing has actually tried to verify the proposed solutions. Instead, a framework's flexibility [122] and/or its similarity to psychological concepts [124] have been often cited as proof of its power.

Since trust is certainly a complex issue, validating systems that attempt to incorporate human trust might be far from trivial. Maybe a system does not possess enough information in order to reach the same conclusion as human does. Even if the systems would get enough data through user solicitation or observation, the usefulness of such a system would probably depend largely on the subjective attitudes of each user, rather than actual system performance. Perhaps, there exists a fundamental incompatibility between the human notion of trust and the computational processes that try to mirror them. Should a system not yield a result that matches human perception, it might simply indicate a lack of consistency of the user (who might feed conflicting information into the system), rather than a system design problem.

Given the difficulties associated with validating the computation trust, it is proposed here that while solutions to very specific problems might be able to benefit from a very restricted computational notion of trust, it generally works better by supporting the human-based trust decision process, i.e. by providing relevant information on demand but leaving it to the individual's state of mind whether to trust or not instead of trying to mimic it. This, in essence, is the approach taken by some privacy infrastructures (e.g. PawS [55]), and is further advocated by this author's work on privacy protection solution that will be proposed in the following chapters.

3.4 Privacy-Respecting Solutions

In addition to individual privacy techniques and mechanisms as discussed in the previous section, some context-aware prototypes and ubiquitous systems have attempted to provide integrated solutions that combine various privacy techniques and mechanisms in their architecture. Table 3.1 below summarizes the features of some salient research work with respect to the privacy techniques and mechanisms they have chosen to include. The "Star marks" highlight key strengths of each presented privacy solution. The table is not meant to be an accurate description but gives readers a rough comparison of the presented privacy work, as well as a hint of current privacy research status in the field of context-aware computing.

The surveyed work are among the most salient privacy research in the field of context-aware ubiquitous computing, however, the author's study of them has indicated that they only addressed a small subset of privacy needs and challenges faced in context-aware paradigm. Some major limitations in approaches taken by them are discussed as follows.

First of all, most of the solutions did not tackle the challenge of the distributed existence of context information. Solutions like PawS [55], e-Wallet [17], Gaia [84], and CoBrA [46,102] were designed for SmartSpace environments (e.g. an enterprise, a campus, a meeting room, etc.). The SmartSpace environments are typically closed and homogenous, and often have a centralized unit holding all the information within the environment and controlling information access. Other more generic solutions (e.g. Confab [61], WASP [94], PACE [44], User-adaptive system [62], Geopriv [76], and CoPs [104]) proposed for context-aware middleware and service platforms, sometimes implicitly, assumed that context information is neatly partitioned into repositories that are under the control of its owner [61], or into a centralized control component and a database in the middleware architecture [44, 62, 76, 94, 104]. The constrained system environments and simplifying assumptions make existing solutions at best not scalable,

and at worst incapable of tackling the challenges of context-aware ubiquitous computing environments. It is clear that solutions need to be able to deal with information that is accessible, but preferably not centrally controlled by any party. Furthermore, while heterogeneous types of context information exist, many solutions like [15, 76, 77] are only concerned with location privacy, while other more generic solutions such as Confab [61] were largely applied to location issues rather than to other forms of context information, although they were designed to address generic types of context information.

Secondly, quite a few surveyed solutions (e.g. Aura [16], Gaia [84], User-adaptive system [62], and Geopriv [76]), and many ubiquitous computing systems that have not been included in this author's survey, are not user-centric but system-centric. In other words, these solutions are concerned mainly with access control and security mechanisms appropriate to the information that is kept within their systems, but did not provide features to support active participation of individuals in controlling their own information Other more user-centric solutions, such as Confab [61], PawS [55], E-Wallet [17], CoBrA [46], WASP [94], PACE [44], Myles's [15], and CoPs [104], attempted to take individual privacy requirements into account when making information disclosure decisions, but their efforts to develop flexible mechanisms to facilitate relatively unobtrusive user participation is very limited. In particular, PawS [55] discussed privacy notification, feedback and consent mechanisms in a SmartSpace environment, but no actual implementation has been made. Confab [61] and e-Wallet [17] developed simple user interfaces to communicate to users information disclosure and to allow users to express privacy requirements. The interfaces, like those in conventional data management environments, were presented as forms with predefined layout and options. Although the options took some context information (such as time, user location, etc.) as disclosure conditions, such a fairly simple approach has limited use where a user's willingness to share personal information may depend in part on the user location, recent and current activities, and may change over time. More interesting user privacy management approaches were presented in e-Wallet [17] and CoBrA [46].

Table 3.1 *The summary of main features of surveyed work*

	Myles [15] 2003	**Confab [61] 2002-2005**	**PawS [55] 2001-2005**	**Adaptive System [62] 2003**	**E-Wallet [17] 2003**	**WASP [94] 2003**	**IETF Geopriv [76] 2004**	**CoBrA [46,102] 2004**	**AURA [16, 77] 2003-2007**	**PACE [44] 2004-2005**	**CoPS [104] 2005**
Anonymity & Pseudonymity	Pseudonym			√★ Mixed Network			√★ Pseudonym				
Access control (user's participation Y/N)	√★ Y	√ Y	√ N	√★ N	√★ Y	√ Y	√ N	√Y	√★ N	√ Y	√★ Y
Encryption and Secure Communication	√(digital signature)		√	√★			√★				
Privacy Meta Data	√★data collecting policy based on P3P	√★Privacy Preference	√★data collecting policy based on P3P		√privacy preference	√★data collecting policy and privacy preference P3P		√Privacy preference		√Privacy preference	√★Privacy preference
Computational Trust											
Generic or specific Context	Location	Generic	Generic	Generic	Generic		Location	Personal Profile, Location	Location		
Application environments	Middleware	Middleware	SmartSpace		e-Campus	Service platform		SmartSpace(e-meeting room)		Middleware	Middleware

They proposed to employ agent technology to minimize user interactions when dealing with requests for users' sensitive information. In both approaches, rule-based logic engines were developed to facilitate automated control of information disclosure. However, the two solutions can only evaluate very simple policies and suffered from the informal way that privacy policies were defined.

Thirdly, only limited work has been reported to develop privacy expression languages and mechanisms to facilitate individual privacy expression. As discussed in the privacy challenges in section 3.2, individual privacy requirements in a dynamic context-aware environment are likely to consist of a complex set of rules, in response to various situations and changes over time. This requires a shared model for expressing privacy policies between a user and an environment and also among users in the environment. However, current solutions appear unable to address this requirement. For instance, Confab [61] provided a programming model for its privacy meta data (i.e. Privacy Tag), by which users could declare what they want information collectors to do with their information. However, no definition of the syntax and semantics of such declaration has been made. PawS [55] and Myles's [15] followed the widely-accepted P3P practice and let information collectors state their data collecting practice, so as to enhance users' trust in disclosing their personal information; but they were not concerned with developing appropriate language specification to allow information owners to express their privacy requirements. Other P3P-based approaches such as WASP [94] demonstrated a simple modification of the P3P-based preference formulation language APPEL [105], for personal preference expression. However, the author's study of APPEL has identified that it is insufficient as a preference language to express semantically-rich privacy preferences towards dynamic context-aware environments. A thorough review [106] of APPEL, conducted by IBM Almaden Research Center, has summarized major problems with the design of APPEL and suggested that the problems cannot be solved without a complete redesign of the language. Other policy-based methods for privacy policy declaration were presented by CoBrA [46], e-Wallet [17] and CoPS [104]. In particularly, CoBrA [46] and

e-Wallet [17] employed semantic web techniques to support a more formal representation of personal privacy requirements. However, the policy definition in both cases has been done in an ad-hoc manner. The policies were only applied to their specific system environments and applications, i.e. e-meeting and e-campus respectively. Neither of them supported privacy preferences involving the purpose, the recipient, or the retention aspects of data collection that have been specified in privacy standards like P3P. To the best of the author's knowledge, no attempt have been made in privacy research in the field of ubiquitous computing to develop formal language specification and associated semantic reasoning mechanisms to represent and enforce context-dependent privacy requirements. The Privacy Policy/Preference Language proposed in Chapter 5 is aimed to fill in this gap.

3.5 Privacy Requirements and Design Guidelines

In this section, the author explores individual privacy concerns over privacy and privacy protection in context-aware ubiquitous computing environments, and examines existing legal and regulatory principles. The investigation work leads to the delivery of the privacy design guidelines that will direct the development of the author's privacy solution presented through Chapter 4 to Chapter 6.

3.5.1 Individual Privacy Concerns in Context-aware Computing Environments

Section 3.1 discussed the definitions of privacy and its history with technology advance. Moving further from general definitions of privacy, this section attempts to qualify, within the scope of context-aware paradigm, the phrase *personal privacy,* by exploring individual concerns over privacy and privacy protection. Research prototypes and context-aware ubiquitous systems [61, 17, 93, 94] as well as different design guidelines

[13, 18, 64, 107, 108, 109] for privacy-sensitive systems have been examined. A brief summary of the literature review is presented as below.

On the one hand, users in context-aware ubiquitous computing environments desire simple and appropriate levels of control over information disclosure, they want feedback of information disclosure with respect to recipients, purposes and conditions, and they have concerns over long-term retention of personal data and potential divulgence to third parties; This is a largely similar situation to conventional computing systems. In addition, according to [61], many people ask for system ability to override privacy needs in emergency situations.

On the other hand, there are two key privacy needs that have not been adequately addressed by researches in conventional computing systems and in ubiquitous computing. The context-aware paradigm requires a broad applicability of plausible deniability and ambiguous disclosure to avoid potentially embarrassing situations, undesired intrusions and unwanted social obligations [18].

- People desire *plausible deniability* whereby the potential observer cannot determine whether a lack of disclosure was intentional. According to [18], mobile phones serve as a good example, in that if a person does not answer a call, it could be a technical reason (e.g. being outside of reach) or for social purposes (e.g. not wanting to talk to the caller right now).
- The system ability to allow *disclosing ambiguous information* is desired, as users' privacy preferences are often not black-and-white but rather involve different levels of accuracy or inaccuracy. According to [17], the ambiguous disclosure is often applied in two ways, either abstracting away some details of information, or providing false information on purpose, such as using pseudonyms to hide a real identity. Allowing individuals to control the accuracy or precision of the information provided in response to different queries under different conditions is indeed a challenging job, as "technical systems are notoriously awkward at

supporting social nuance" [107].

Table 3.2 summarizes the capabilities of the surveyed work that have been presented in the last section, with respect to meeting the individual privacy requirements identified above. The "Star marks" highlight the key strengths of each presented privacy solution. It is shown that Confab presents a significant work in terms of meeting most of the privacy requirements that were identified above, and to the author's knowledge, few privacy work in the context-awareness is as advanced. The table also indicates that the privacy solutions have addressed only a small subset of the privacy needs faced in context-aware paradigm, and it seems that personal privacy needs on plausible deniability and handling emergency, remain largely unsatisfied. In addition, significant privacy issues with regard to unobtrusive mechanisms to empower individual active participation with control over the disclosure of their information, including getting notice of relevant information disclosure, feedback, and explicit consent have not been adequately researched.

Table 3.2 *The summary of capabilities of surveyed work with respecting to satisfying the privacy needs*

	Myles [15] 2003	Confab [61] 2002-2005	PawS [55] 2001-2005	Adaptive System [62] 2003	E-Wallet [17] 2003	WASP [94] 2003	IETF Geopriv [76] 2004	CoBrA [46, 102] 2004	AURA [16, 77] 2003-2007	PACE [44] 2004-05	CoPS [104] 2005
Users' participation in disclosure control (giving consent and choice)		√★ via user interface			√★via user agent and interface	√ via user agent					√
Getting feedback	√★via data collecting policy	√	√★via data collecting policy			√★via data collecting policy					
Mechanisms to ease concerns over long-term Info Retention	√★ via declaring p3p-based collecting policy	√★ via expressing privacy preference	√★ via policy-based database management			√★via expressing p3p-based privacy preference					
Plausible Deniability		√									√
Ambiguous Disclosure		mentioned for location data, but no actual implementation	mentioned for location data, but no actual implementation		√★			√★		√ via context information modeling	√★
Handling Emergency		√									

3.5.2 The Fair Information Practice Principles

Legislation and regulatory efforts also set requirements for providing individual privacy. Two of the most influential pieces of privacy legislation—the US Privacy Act of 1974 [110] and the EU Directive 95/46/EC of 1995 [111]. The two legislation efforts have the goal to ensure that personal privacy is safeguarded when new information technology applications are developed. Although the effect and effectiveness of privacy protection through legislation efforts remains to be seen, principles underlying the two important privacy legislations are worth exploring.

The notion of "Fair Information Practices" was created during the enactment of US Privacy Act of 1974, and has been incorporated into all major pieces of privacy legislation worldwide and continues to shape privacy legislation throughout the world [112]. Although carrying no legal obligation, the Fair Information Practices suggest a baseline for privacy rules that all system design (either technological or organizational) ought to consider.

The principles of Fair Information Practices, which in turn are based on work by Columbia University political economist Alan Westin, are summarized by [14] as follows:

"

1. *Openness and transparency: there should be no secret record keeping, this includes both the publication of the existence of such collections as well as their contents.*
2. *Individual participation: the subject of a record should be able to see and correct the record.*
3. *Collection limitation: data collection should be proportional and not excessive compared to the purpose of the collection.*
4. *Data quality: data should be relevant to the purposes for which they are collected and should be kept up to date.*
5. *Use limitation: data should only be used for their specific purpose by authorized*

personnel

6. *Reasonable security: Adequate security safeguards should be put in place, according to the sensitivity of the data collected.*
7. *Accountability: Record keepers must be accountable for compliance with the other principles.*

"

In the early 1980s, the Organization for Economic Cooperation and Development (OECD) took up those principles and issued "The OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data" [113], which described eight practical measures aimed at harmonizing the processing of personal data in its member countries. Even though the OECD principles carried no legal obligation too, they nevertheless constituted an important international consensus that substantially influenced national privacy legislation in the years to come [114]. A summary of the OECD guidelines can be found in Appendix C.

In 1996, The EU Directive 95/46/EC subsumed and refined the fair information practices described above, and added the notion of *explicit consent* ——personal data may only be processed if the user has unambiguously given his or her consent [111]. This particularly disallows all types of data collection (except for when required by law) and requires a case-by-case explicit consent by the data subject.

Taken together, the three sets of guidelines above could be summarized as six basic principles:

Instructing data collectors in data collection practice:

- *Openness:* there should be no secret data collection and record keeping. The existence of such collections and recordkeeping, as well as their contents should be made public.
- *Data Minimization:* Personal data should be relevant to the purposes for which they are to be used, and, to the extent necessary for those purposes, should be accurate,

complete and kept up-to-date.

- *Data Security:* Reasonable measures need to be taken to secure (both technically and operationally) the data from unauthorized access, modification, disclosure, and misuse (i.e. use not for intended purpose).

Instructing users' participation in data disclosure practice:

- *Notice:* There must be a way for individuals to find out what information about them is in a record and how it is used, and third parties other than the original collector who will have access to the data.
- *Choice and Explicit consent:* Individuals should be allowed to choose not to have data collected, as well as the level of detail of the collected data. Systems should always get the consent of individuals before their data collection, and before information about them obtained for one purpose is used or made available for other purposes.
- *Data access and control:* Individuals should be allowed to see what personal information is held about them, to correct errors, and to delete the information if desired.

3.5.3 Privacy Design Guidelines

The author intends to incorporate the principles of Fair Information Practices into the design of the privacy solution proposed in this work, in addition to overcoming the limitations of existing approaches as identified in section 3.4. Integration of social and legal privacy considerations helps lessen the risks of providing only a shallow and short-lived privacy solution – either because it is incompatible with social realities, or inconsistent with existing legal frameworks.

However, not all the principles of Fair Information Practices could be fulfilled through even rigorous employment of technical mechanisms. As discussed in section 3.2, there

are social and organizational issues that simply cannot be managed by technical means alone. For instance, the Data Minimization principle requires data collectors to only collect data for a well-defined purpose (no "in-advance" storage), only collect data relevant for the purpose (not more), and only keep data as long as it is necessary for the purpose. While technical mechanisms like privacy meta data might be useful in terms of helping data collectors to state their collecting policies and keeping users informed about information disclosure, without social and legal pressure to compel data collectors to comply with their stated policies or announced privacy policies by users, the individual privacy needs may eventually be overridden.

Combining the principles of the Fair Information Practice and the technical realities, the author proposes four design guidelines that the technical solution proposed in this work could and does attempt to accomplish.

- **Notice:** Providing unobtrusive ways to communicate the information collecting to users, including declaring collected data, its intended use, duration of valid use, and the indication of third parties other than original collectors who will have access to the data.
- **Choice and Consent:** getting the consent of users before data collection, and before their information obtained for one purpose is used or made available for other purposes. In order to make the consent a viable option, more than just "take it or leave it" dualism, appropriate selection mechanisms are required to empower users to choose information disclosure with the right people and services, in the right situations, and at the right level of detail, in addition to the ability to choose not to have data collected.
- **Preference Access and Management**[9]**:** provisioning unobtrusive ways for users to manage their privacy preferences, including to see what preferences are held, to update new preferences, and to delete existing preferences if desired.

[9] The solution proposed by the author will focus on access to individual privacy preference information, instead of other types of personal information that are likely distributed and not controlled by the person who is referred in the information.

- **Data Security:** protecting information confidentiality, integrity and non-repudiation, in particular, an access control model and mechanisms against unauthorized access, and encryption and cryptographic tools to secure data transmission.

Fulfilling the four design principles helps address some of individual privacy concerns that have been identified in the previous section 3.5.1. In particular, the Notice principle is a direct correspondence to privacy need for feedback. It also helps to some extent ease individual concern over long-term retention of personal data and potential divulgence to third parties, as it provides social visibility to prevent abuses. The Choice and Consent principle as well as the Preference Access and Management principle are key elements to help fulfill users' desire for control over their sensitive information. The Data Security principle helps prevent potential divulgence of sensitive information to unwanted third parties when it is disseminated in ubiquitous computing environments. The four principles summarized above will govern the development of the privacy-respecting solution that will be presented through Chapter 4 to Chapter 6.

3.6 Chapter Summary

This chapter has provided an understanding of issues involved in protecting individual privacy in dynamic context-aware ubiquitous environments. It explored the reasons why it is challenging to protect privacy in such environments, discussed possible technology offers and mechanisms that can be and are employed to preserve privacy, and investigated existing privacy solutions in the field of context-aware ubiquitous computing. The most salient aspects of the background technologies and relevant work have been presented. The analysis of the most salient privacy work in the area has indicated that only a small subset of the privacy needs and challenges have been moderately addressed, and solutions for adequate privacy protection in context-aware

paradigm are still awaited. This justifies the author's effort to develop an adequate privacy protection approach to tackle the challenges and overcome the limitations of existing solutions. At the end of this Chapter, the author developed privacy design guidelines by exploring user privacy concerns in context-aware environments, and examining existing legal and regulatory principles. The guidelines will direct the development of the author's privacy solution that will be presented through Chapter 4 to Chapter 6.

Chapter 4: A Distributed Privacy Protection Model and An Intelligent Agent Technology for Privacy Protection

This chapter presents the author's proposal of a distributed privacy protection model for context-aware ubiquitous computing environment and work on an intelligent agent technology to overcome the limitations of existing privacy solutions and technical challenges that have been identified in previous chapter (section 3.4 and 3.5 respectively). It sets out by plotting privacy protection scenarios in an envisioned ubiquitous city. Discussions based on the scenarios lead to an important understanding of ubiquitous systems as a composite environment as well as a full analysis of roles played by various parties involved in preserving personal privacy in such an environment. The chapter then focuses on introducing the distributed privacy protection model, and describing the architectural design of the Privacy Agent technology, its key components and working logic. The reasoning behind the key design considerations will be discussed. The final part of this chapter identifies limitations of the proposed work, mainly in terms of security threats and trust concerns, and discusses possible safeguard or mitigation measures. Detailed design and implementation of some key parts of the author's proposal will be presented in Chapter 5 through Chapter 7.

4.1 Privacy Protect Scenarios in Context-aware Ubiquitous Computing Environment

This section depicts basic notions of preserving privacy in context-aware ubiquitous computing environments and helps project clearly what exactly the proposed intelligent privacy agent will achieve. Many of the scenarios are constructed to be in direct correspondence to the personal privacy needs and the privacy challenges identified in

chapter 3.

Imagining a wireless-networked city –eLondon– offering various context-aware ubiquitous computing services. The services may be available citywide, e.g. a location tracking service provided by local mobile operators, or be provisioned only for people within a certain space, such as a personalized shopping-guide application in each shop.

Alice is a foreign tourist visiting the city and carries her smart phone (that presumably integrates the functionality of a mobile phone, personal digital assistant or other information appliances) in order to use context-aware ubiquitous computing services. The smart phone serves as a personal assistant and provides Alice an interface to specify her privacy preferences. The privacy preferences are uploaded to and stored at Alice's Privacy Agent (PA) residing somewhere on the network.

It is assumed that Alice has specified that any services or applications can use pseudonyms stored on her smart phone to deliver personalized services without alerting her, while any services or applications requiring her real identity and exact location must have her explicit consent.

Once Alice steps into e-London city, her smart phone registers with a local mobile carrier under a roaming service agreement between her mobile operator at home and the local mobile carrier. Context-aware service providers built upon the local mobile network soon advertise two services:

I. *A Who-near-me* service which promotes a friendly atmosphere to tourists by notifying them when their friends, alumni, favorite movie stars, etc. are by chance traveling the same city. This service is provided by the local mobile operator and is optional to use. To use the service, tourists are required to disclose their real identity, social group and location information. The social group information defines social categories, e.g. friends, colleague, alumni, etc., that are pertinent to a particular person. Used together with a real identity, the

social group information ensures that a tourist's location information is shared with the right people. Also, tourists could choose to disclose their location information at different levels of granularity, e.g. at *city level*, *district level*, or *street level*.

II. *An Emergency Notification* service which provides a city-wide location-tracking capability to keep locating tourists and which notifies them of any unexpected emergencies happening nearby, such as a building fire. The *Emergency Notification* service is provisioned by local authorities, and its use is mandatory. Being mandatory does not imply that tourists have to accept a service offer; it means that when tourists accept a service offer and enter a service agreement with service providers, they cannot choose to opt out from the service whenever they want. This is equivalent to the application of security cameras in a supermarket, and is in contrast to the *Who-near-me* service whose use is optional. The *Emergency Notification* service asks for a tourist's real identity, exact location and personal contact (such as his/her home phone number, postal address) to function. The personal contact information is used to notify relevant people if tourists are caught up in an emergency or any injury happens.

The service advertisement not only states what are the context-aware services that Alice will benefit from, but also states the accompanying data collecting policies of the services; for example, it specifies the data collectors, the requested information with desired level of granularity, the intended use, the expected duration of use, the potential third parties, etc. Alice's Privacy Agent reads the collecting policies, and then compares them with her privacy preferences. Conflicting interests are detected since the *Emergency Notification* service asks for Alice's exact location to function, and since the *Who-near-me* service offers information granularity levels for user disclosure choices. The Privacy Agent then notifies Alice (through her smart phone) of the privacy conflict and choices, and waits for her approval or rejection of the service offer. Note that in the case no conflict of interest is detected, the Privacy Agent will not intrusively notify or alert Alice.

Alice then finds that the *Who-near-me* service is interesting, but since it is her first time to use such a service, she has concern that others might be able to infer what she is doing by looking at her exact location. She decides to share her location information only at the city level. She also finds the *Emergency Notification* service is necessary because of recently soaring terrorist threat. She then replies to her Privacy Agent that she would like to accept both service offers and accepts the compromise of her wish for privacy (i.e. disclosing her location information, social contact together with real identity).

III. Alice then walks into a souvenir shop where a *Shopping Guide* service——a personalized advertisement service based on tourist's personal profile (e.g. gender, age, occupation, purchase history, etc.) —— is offered. Alice's Privacy Agent recognizes the need for a unique identity to use this service, but continues to respect Alice's privacy by offering a pseudonym in place of her real identity. Only when Alice checks out, Alice uses her credit card with real identity information for payment.

4.2 A Composite Context-Aware Ubiquitous Computing Environment and Various Parties in Preserving Personal Privacy

Based on the privacy protection scenarios above, the author introduces in this section a novel understanding of ubiquitous computing systems as a composite environment where a distributed privacy protection model and intelligent agent technology for privacy protection lay on. The notion of a composite ubiquitous computing environment indeed presents a different view from existing solutions. As identified in section 3.5, the existing solutions either focus on constrained SmartSpaces, or are only concerned with

ubiquitous architectures based on specific network systems, such as mobile networks. These solutions seldom take into account heterogeneous types of ubiquitous environments and hardly address different systems all together.

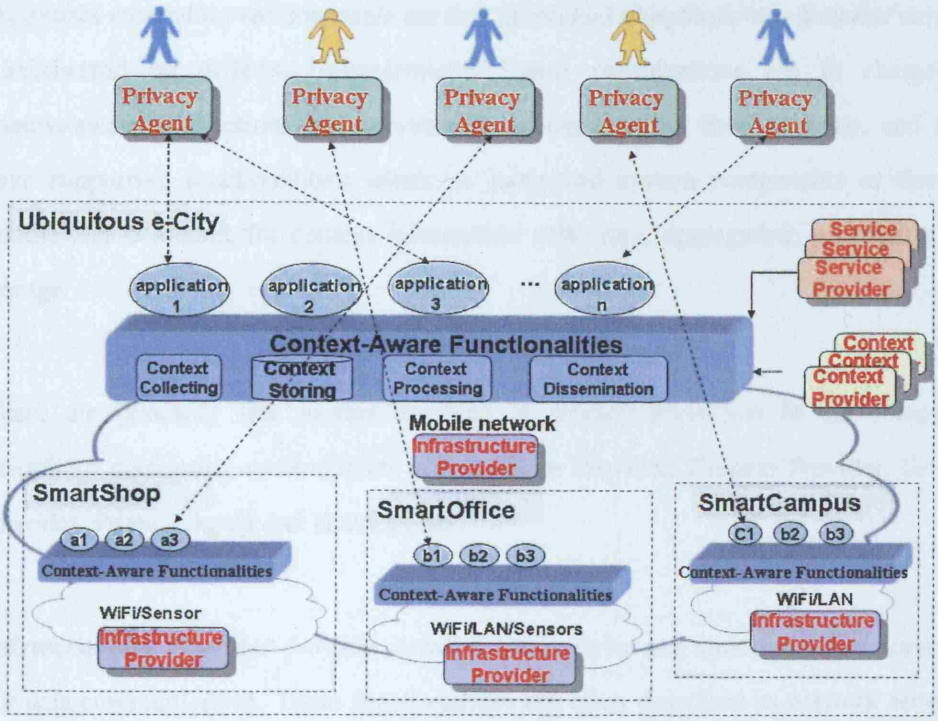

Figure 4.1 *A composite ubiquitous computing environment in an e-City*

As illustrated in Figure 4.1, a composite ubiquitous computing environment (such as the e-London city depicted in the user scenario) is divided into multiple sub-domains, each of which represents a ubiquitous system and which are outlined with blue broken lines. The sub-domains could either be constrained SmartSpace systems in shop, office, campus, etc, or more open environments like middleware-based context-aware service platforms that are built upon the incumbent mobile network. A composite ubiquitous computing environment might itself be seen as a sub-domain of another composite environment on a larger scale (e.g. the whole country.)

Multiple ubiquitous system domains may have their underlying infrastructure interconnected or overlapped, for instance, a wireless network in a smart office domain

may be connected to the Internet that probably also links other SmartSpaces; mobile connectivity in a ubiquitous city also includes sub-domains of SmartSpaces within its coverage. However, the important assumptions underlying the notion of composite ubiquitous computing environments are that individual ubiquitous sub-domains may be administered by different organizations. These organizations are in charge of context-aware applications and services provisioned within their domains, and they have supporting functionalities, either as integrated system components or through middleware solutions, for context information collecting, aggregating, processing and storage.

There are primarily five parties involved in privacy protection in the composite ubiquitous computing environment: Infrastructure Provider, Context Provider, Service Provider, Privacy Agent and Human User.

Infrastructure Provider provides network connectivity and underlying functionalities for data communication. These functionalities are often described as network services, in contrast to application-level services. In the privacy protect scenarios, the role of the *Infrastructure Provider* is played by mobile operators in the ubiquitous city, and by business and institutional organizations who run Wi-Fi or sensor networks either in wide-area (e.g. "cloud" across the e-London city) or just in a premise.

Context Provider *is* responsible for providing context information to be used by various consumers. The consumers include *Service Providers* that provision context-aware applications to end users, and the Privacy Agent (in the author's proposal) that attempts to take advantage of users' context information to make appropriate information disclosure decision.

Since context information requested by various consumers may be at different levels of accuracy and abstraction, *Context Providers* have capabilities to conduct the process of

aggregating, inferring, and correlating of information before it is desired by requesters. The role of *Context Provider* separates the context provisioning capability from the integration of service provision and management of context-aware applications, and facilitates a quick deployment of context-aware applications by shielding them from the complexity introduced by handling contextual information.

Service Provider of context-aware applications is often business or institutional parties that utilize networking functionalities provisioned by *Infrastructure Providers* and context information provided by *Context Providers*, to support quick development and deployment of context-aware applications to end users. Examples of *Services Provider* in the privacy protection scenarios are local mobile operators that provision *Who-near-me* service, local authorities that offer *Emergency Notification* service, and individual shops that run *Shopping Guide* service.

Human Users are central to protecting their sensitive information in the composite ubiquitous computing environment. In the author's proposal, every one who wants to preserve privacy each has one **Privacy Agent** that stores his/her privacy preferences. As illustrated in Figure 4.1, individuals interact with context-aware applications provisioned in various ubiquitous systems domains via their *Privacy Agent.* The Privacy Agent is able to act without continuous intervention by individuals, and is designed to assist individuals in disclosing their sensitive information to dynamic context-aware environments with relative ease. The information disclosure control primarily includes privacy feedback (i.e. notifying individuals of any relevant information disclosure) and privacy management (i.e. allowing them to express their privacy preferences and control their privacy levels). Section 4.4 will present the design of key components of a Privacy Agent and its working logic.

Discussions

Roles of Infrastructure Provider, Context Provider, and Service Providers may be

played by the same business or institutional organization. For instance, *Infrastructure Provider* may itself act as *Service Provider* of context-aware applications (e.g. the *Who-near-me* service) provisioned in its infrastructure, in addition to opening the infrastructure to allow third party service, e.g. the *Emergency Notification* service offered by local authorities in the privacy protection scenario. In the SmartShop domain, as described in the privacy protection scenarios, individual shops play the combined role of *Infrastructure Provider*, *Service Provider* and *Context Provider*.

The fact that multiple roles are played by a single organization has important implications for preserving privacy in ubiquitous computing environment. In the ubiquitous computing paradigm, context information relevant to human users is dispersed in the environment; for instance, a person's location information could be collected by mobile operators, his contact information may be stored in his mobile phone device, and real identity is held by the person themselves. There is unlikely to be a centralized control unit solely responsible for information protection in such an environment. Human users are owners of their sensitive data, but are not necessarily the only information holder. The Context provider, Service Provider and Infrastructure Provider can all be information holders when personal information is generated and consumed by, or bypasses them. If the roles of Context Provider, Service Provider and Infrastructure Provider are played by a single organization, a clear separation of accountability of the different parties involved in privacy protection is harder to establish. To tackle this problem, the author proposes a distributed privacy protection model in the composite ubiquitous computing environment, which will be elaborated in the next section.

4.3 A Distributed Privacy Protection Model in a Composite Ubiquitous Computing Environment

The author introduces in this section a distributed privacy protection model that separates where the information disclosure decision is made from where information is generated, consumed, and held and thus where the information disclosure decision is actually enforced. The distributed privacy protection model takes into account the dispersed nature of context information in dynamic context-aware ubiquitous computing environments, and overcomes limitations of existing solutions, which are based on many simplifying assumptions, e.g., they only consider location information, or assume that context information is neatly partitioned into repositories that are under the control of a single user. In addition, separating a privacy decision process from an enforcement process allows multiple parties to participate in the control of information disclosure with clear accountability. For instance, in an enterprise environment, system administrators may enforce an enterprise-wide security policy while taking individual privacy preferences into consideration.

Figure 4.2 presents the distributed privacy protection model and information access flow according to the privacy protection scenario depicted in section 4.1. The author follows the Common Information Model (CIM) specification (IETF RFC3198 [125]) and term the policy decision point as Privacy Policy Decision Point (PPDP), and where information disclosure decision is actually enforced as Privacy Privacy Enforcement Point (PPEP). In addition, this work introduces a new entity, Privacy Policy Administration Point (PPAP), to represent system entities that create privacy policies and preferences.

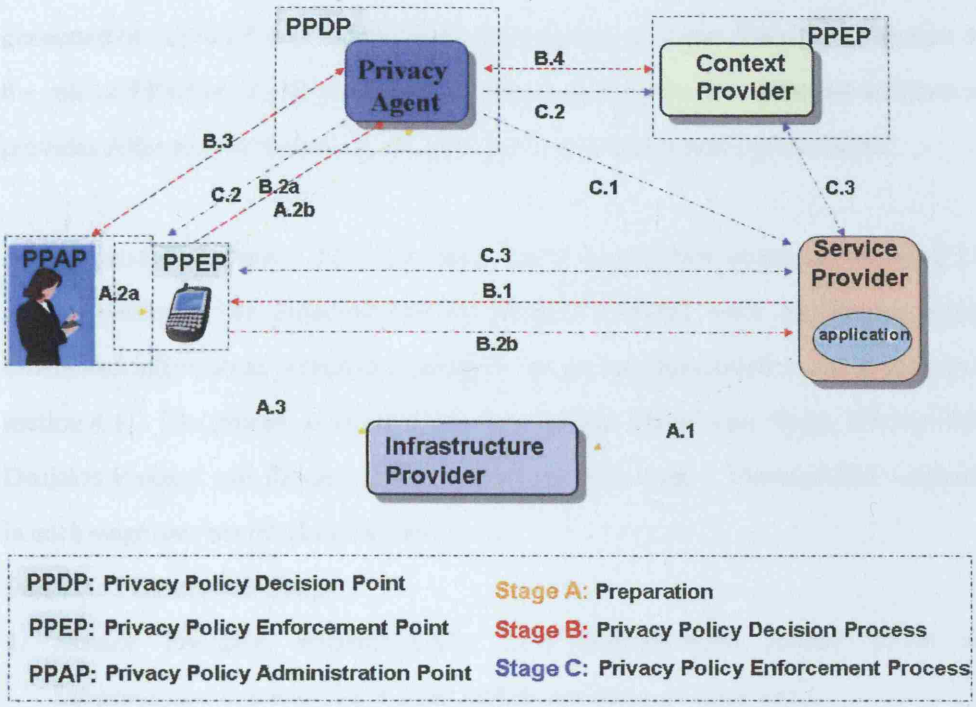

Figure 4.2 *A distributed privacy protect model and message flow sequence*

The idea of a Privacy Agent is a new concept introduced in this research, which plays a key role in handling privacy interactions on behalf of human users. As illustrated in Figure 4.2, the Privacy Agent acts as PPDP in the proposed distributed privacy protection model, it evaluates data collecting policies received from *Service Provider* and makes decision about how a person's sensitive information is disclosed, to whom and at which level of granularity. Note that, the Privacy Agent does not hold the information nor enforce the information disclosure decision. The sensitive information is protected separately by multiple PPEPs where it resides. PPEPs act as a main divider between information requesters and requested information. PPEPs grant information access only if the PPDP (i.e. Privacy Agent) agrees the information disclosure. Depending on where information is generated and held, entities that play the role of PPEPs in the distributed privacy protect model include human users who hold their own information (e.g. real identity), devices that people carry and where personal contact information may be stored, Context Providers where people location information is

generated or collected, etc. In the privacy protection scenario described in section 4.1, the role of PPAP is played by Alice's smartphone. It serves as a personal assistant and provides Alice an interface to create, edit, and update her privacy preferences.

Also presented in Figure 4.2 is an information access flow sequence, showing how various parties in the proposed privacy protection model work together to perform distributed information protection, based on the privacy protection scenario depicted in section 4.1. The process is divided into three stages: Preparation Stage, Privacy Policy Decision Process, and Privacy Policy Enforcement Process. Message flow sequences in each stage are described as follows:

Stage A: Preparation Stage:

1. *Service Providers* register (A.1) their context-aware service offers with *Infrastructure Provider*, i.e. a local mobile carrier in e-London city.
2. Alice through her smart phone sets (A.2.a) her privacy preferences and updates them to her Privacy Agent (A.2.b). It is proposed that the Privacy Agent resides somewhere on the network for connectivity and power-saving reasons, and can be reached whenever needed by users via, for example, the Internet access.[10]
3. Once Alice steps into e-London city, her smart phone registers (A.3) with the local mobile carrier under a roaming service agreement between her mobile provider at home and the local mobile carrier.

Stage B: Privacy Policy Decision Process

1. A context-aware application sends (B.1) Alice's smartphone a service offer and states its collecting policy for Alice's personal data.
2. Alice's smart phone replies (B.2.b) to the context-aware application with the address of Alice's Privacy Agent, and in the mean time forwards (B.2.a) the data collecting policy to Alice's Privacy Agent, if privacy protection is required by Alice.

[10] The author's proposal does not take into account any specific access technologies for the Privacy Agent. This issue is left to decisions by the implementers of the Privacy Agent in different network environments.

3. Alice's Privacy Agent parses and compares the data collecting policy with privacy preferences set by Alice to make information disclosure decisions. It has inference engines and reasoning mechanisms to enable automated disclosure decision making. But in some case, explicit consent (B.3) from users about information disclosure is necessary. This is exemplified in the privacy protection scenario, where a conflict of interest occurs between the data collecting requirement of the *Emergency Notification* and Alice's privacy preferences in terms of her real identity and location information.
4. To make appropriate information disclosure decisions in response to Alice's context changes, the Privacy Agent must be fed with contextual knowledge about her. It acquires such knowledge by consulting (B.4) with context information providers. Multiple Context Providers exist, which include context-aware applications that Alice's is currently subscribing to, or functional applications embedded in her smartphone, such as a personal calendar, etc.

C. Privacy Policy Enforcement Process

1. Once a positive information disclosure decision is made, the Privacy Agent generates a privacy agreement (termed as a *Privacy Contract* in this work) based on the data collecting policy received and Alice's privacy preferences, and encloses the agreement into a pair of credentials. One credential will be sent (C.1) to context-aware application that made the information request, while another will be distributed (C.2) to each PPEP that holds the requested information. Take the *Who-near-me* service for example, one credential will be sent to local mobile carrier where Alice's information location is collected, and one to Alice's smartphone where her social group information may reside.
2. The context-aware application then renders (C.3) its credential to PPEPs for information access. The way an individual PPEP enforces the Privacy Contract received from the Privacy Agent, together with its own access control policies, is beyond the discussion of the privacy protection model proposed in this work.

4.4 Privacy Agent Components and Working Logic

In the author's proposal, the Privacy Agent handles interactions with context-aware applications on behalf of individuals. Human users, who want to preserve privacy, each have one Privacy Agent that stores his privacy preferences. The Privacy Agent has two major functionalities: on the one hand, it mediates privacy-related interactions between individuals and data collectors, including notifying them of relevant information disclosure and negotiating on behalf of individuals with data collectors in accordance with their privacy preferences; on the other hand, the Privacy Agent serves as a continuously running service that can be contacted and queried by users anytime, allowing instant access and adjustment to privacy preferences. This section describes a high-level architecture design of the privacy agent, its key components and working logic. Reasons behind the design will also be discussed.

4.4.1 Key Components of Privacy Agent

In context-aware environments, information used to characterize privacy aspects of a person could be wide ranging and comes from a variety of sources. It is likely that people will change their privacy preferences over time and in response to context changes. To cope with these issues, the Privacy Agent has inference and reasoning capabilities implanted, in order to automatically compute data disclosure policies under different situations according to the users' initial preference settings. Such capabilities are achieved by two key components: the *Preference Evaluator* and *Policy Evaluator* as illustrated in Figure 4.3.

The *Preference Evaluator* is responsible for detecting conflicts and redundancy of privacy preferences. Since the Privacy Agent in the author's proposal is designed to be a continuously running service that can be contacted and queried by its owner anytime, allowing instant access and adjustment to privacy preferences. Conflict and

redundancy of preference rules may occur when users edit new privacy preferences, or update and delete existing preferences. The *Preference Evaluator* serves as an interface to PPAP, where individual privacy preferences are checked before they are stored in *Privacy Preferences Repository*.

The *Policy Evaluator* serves as an interface to context-aware applications, where data requests with associated data collecting policies are received, and policy evaluation results are returned. It is responsible for parsing and comparing data collecting policies of context-aware applications with a user's privacy preferences that are stored in *Privacy Preferences Repository*.

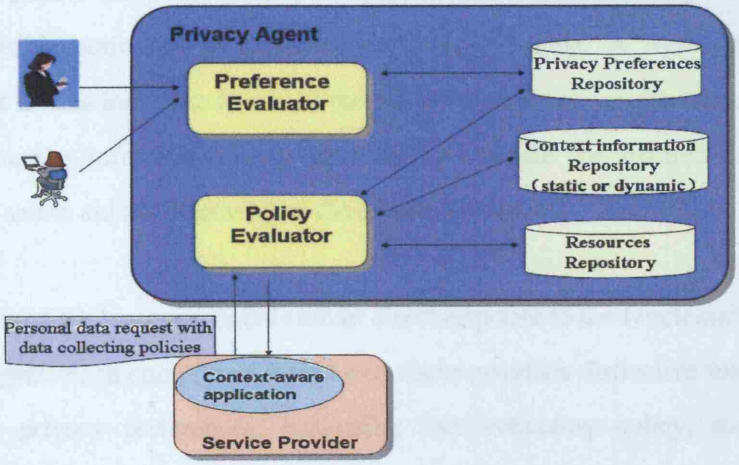

Figure 4.3 *Key components of the Privacy Agent*

To make appropriate information disclosure decisions in response to a user's context changes, the *Policy Evaluator* must be fed with knowledge about the user. The knowledge includes static personal data (such as the user's name, gender, etc), and dynamic context information (such as the user's location, ongoing activities, etc). For instance, a context-aware application requests a person's exact location information but the person has specified her privacy preferences as "her colleagues are allowed to know her location only when she is working, while her parent and close friends can know her location when she is off working." In this case, the *Policy Evaluator* must have

knowledge about whether the person is working (dynamic context) and who belongs to the person's group of friends and colleagues. The *Policy Evaluator* can acquire such knowledge by resorting to the *Context Information Repository* where some context information previously acquired is stored.

In some cases, dynamic context information required to assist policy evaluation might not be available (i.e. not stored in the *Context Information Repository*) at the time context-aware applications make requests. Consequently, the Privacy Agent needs to consult with Context Providers that are either within the ubiquitous environment where the user is, or trusted third-party context providers that are outside the environment but publicly available (such as weather forecast service providers using for predicting a user's possible activities) to perceive the user's context. A *Resource Repository* component is thus introduced to hold the information about various context providers. In addition, the information may be used to evaluate the trustfulness of context providers, and to aid the discovery of the context providers.

The choices of the key components are in direct response to the functional design of the Privacy Agent. Each component introduced above provides distinctive functionality for evaluating privacy preferences, evaluating data collecting policy, storing privacy preferences, storing context information, and assisting context information acquisition. In a real world implementation, some functional components may be implemented by a single physical module. As exemplified in the proof-of-concept implementation in Chapter 7, HP's Jena 2 Semantic Web Toolkit [141] is used to implement functionalities of the *Policy Evaluator* and *Preference Evaluator*, and a XML-based database, Apache Xindice [157], is constructed to store both privacy preferences and context information.

Other agent architectures exist for controlling personal information disclosure in ubiquitous computing environments. In particular, the Context Broker Architecture (CoBrA) proposed by Chen [102] supports context sharing in a SmartSpace

environment and enforces user-defined policies for privacy protection. The architecture consists of four modular components for persistent data storage, context reasoning, context acquisition and privacy protection. Another centralized agent architecture, *semantic e-Wallet*, developed by Gandon and Sadeh [17], acts as a directory of contextual resources for a given user while performing access control according to user-specified privacy preferences. Its architectural components include: a knowledge base, an inference engine, web-services invocation toolkit, and security toolkit.

4.4.2 Working Logic of Privacy Agent in evaluating data collecting policies

The working logic of the Privacy Agent in evaluating data collecting policies is illustrated in Figure 4.4. When the Privacy Agent receives data requests and associated data collecting policies, it first checks with the *Privacy Preferences Repository* and tries to find the matched rules that are applicable to the data requested. If successful, the *Policy Evaluator* loads the rules, and then parses and compares the data collecting policies against the rules one by one. The corresponding rules might not exist for the requested data, in which case the *Policy Evaluator* checks a privacy meta-policy that is set by users and stored in the *Privacy Preferences Repository.*

The privacy meta-policy provides two ways for users to specify a default access policy, an optimistic and a pessimistic one.

- Pessimistic: by default, if no explicit rules are defined to regulate information access, then assuming the access is forbidden.
- Optimistic: by default, if no explicit rules are defined to regulate an information access, then assuming the access is permitted.

Since users can hardly predict all possible scenarios for which they want to grant or deny access, this approach helps effectively reduce the number of rules necessary for

users to manage their privacy. It also enables privacy agents to reason about permissions or forbiddances even when preferences rules are not defined.

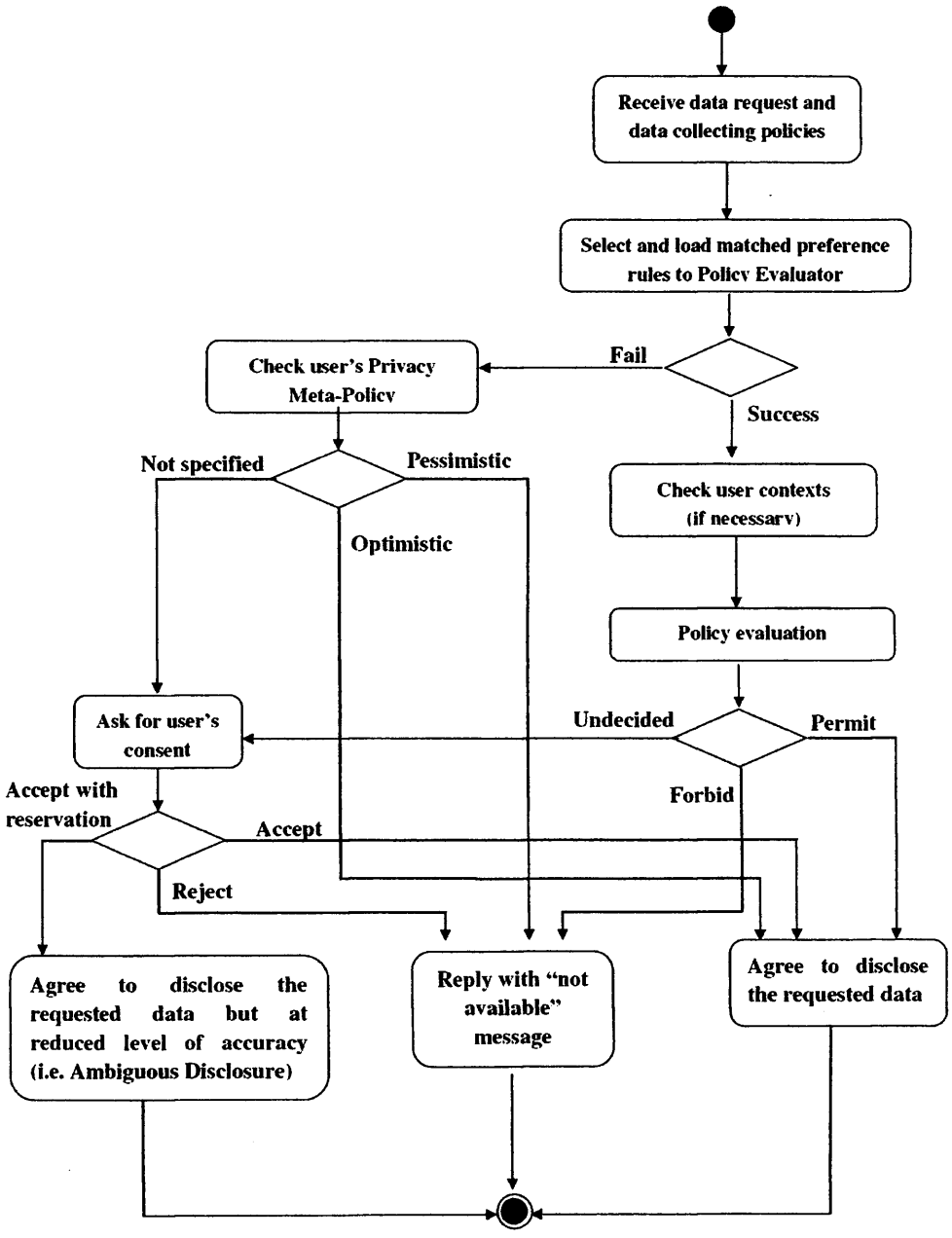

Figure 4.4 *Working logic of privacy agent in evaluating data collecting policies (note: crossing lines do not intersect)*

Rule-by-rule policy evaluation conducted by the *Policy Evaluator* leads to three different results. Ideally, the *Policy* is able to decide whether to permit or forbid the information disclosure. But in more complex cases, rule evaluations might not able to bring about a decisive result. Consequently, the Privacy Agent will notify its owner and ask for her/his consent. This happens mainly in two situations. Firstly, users may specify in their privacy preferences that any requests for certain information (such as real identity, exact location, etc.) must have their explicit consent. In the second situation, users' privacy preferences do not forbid an information disclosure, but the requesting context-aware application offers options in terms of the information granularity levels at which users can choose to disclose. For instance, the *Who-near-me* application (as exemplified in the use scenario in section 4.1) asks for a user's real identity, her social contact and location information, in order to notify her when her friends or family members are nearby. The user may be allowed to choose to share her location information at different levels of accuracy, such as *city level, district level,* or *street level.* The ambiguous disclosure is made available by employing ontological modeling techniques to represent privacy-sensitive personal information in a hierarchy way, so that ontology-based description logic can be used to reason over hierarchical structuring of the information as a means to facilitate semantic matching. The context modeling and inference mechanisms will be described in detail in Chapter 6.

The final note regarding the working logic of the Privacy Agent is that a "not available" message is used instead of a direct rejection when requested data are not allowed to disclose. By doing so, the context-aware applications that make requests and other potential observers cannot determine whether the lack of disclosure was intentional, as it could also due to a technical failure (such as the Privacy Agent does not work, or the user is not accessible at the time). This approach is called *plausible deniability,* and as suggested by [18] is a requirement to avoid potentially embarrassing situations while preserving individual privacy. In reality, it is not uncommon that people may not want others, especially close friends, to know that they do not want to disclose information to them.

4.5 Threat Models and Safeguard Mechanisms

The distributed privacy protection model and privacy agent technology are proposed to tackle technical challenge of protecting dispersed personal information in dynamic context-aware environments, and to overcome the limitations of existing solutions by allowing multiple parties to participate in privacy protection with clear accountability of each party. However, the distributed model also brings some potential security threats and trust concerns when information disclosure decision and enforcement processes are distributed in a ubiquitous computing environment. This section identifies possible security and trust compromise scenarios that should be considered when implementing the distributed model and intelligent agent. Potential safeguard mechanisms are also discussed.

4.5.1 Unsecured Transmission

The first compromised scenario is concerned with message transmission in the distributed protection model. It is assumed that an adversary has access to the communication channel between parties in the distributed privacy protection model and is able to interpret and modify (insert or delete) messages or parts of messages. This middle-man attack can be conducted not only by malicious third parties who have not been involved in the privacy interaction, but also by engaging parties directly involved in the privacy interaction. For instance, the engaging party can use information from a former message maliciously in subsequent transactions. It is proposed that there are basically three methods to conduct the middle-man attack that may be relevant to the proposed privacy protection model.

Unauthorized disclosure of privacy preferences and policies

In the proposed privacy protection model, most messages exchanged consist of information about how to regulate personal information disclosure. Unauthorized access to this kind of information could lead to privacy violation, as a potential adversary may

learn through the information how to circumvent the Privacy Agent to gain unauthorized access. Confidentiality mechanisms are thus required to ensure that the contents of a message can be read only by the desired recipients and not by anyone else who encounters the message while it is in transit.

Message modification

Message modification is a straightforward way to gain direct unauthorized access to sensitive personal information. If an adversary can intercept a message, in particular, a Privacy Contract that contains information on a disclosure decision made by the Privacy Agent, and change its content, it may be able to gain unauthorized access directly. Integrity mechanisms are required to ensure that information has not been altered (including inserting or deleting part of message) since they were originally created in the Privacy Policy Administration Points (PPAPs).

Message replay

Message replay is an attack in which adversaries record and replay legitimate messages among parties in the proposed distributed protection model. This attack is hard to prevent in many security systems, and it may lead to denial of service, the use of out-of-date information or impersonation. Note that encryption of the message does not mitigate a replay attack since the message is simply replayed and does not have to be understood by the adversary.

Mitigation Mechanisms

The author recommends three mitigation mechanisms to tackle the threats models identified above and to ensure secure communication among various parties in the privacy protection model. It is feasible to implement three mechanisms altogether when choosing appropriate security protocols and tools. In the proof-of-concept implementation work in Chapter 7, the author demonstrates the use of SSL protocol and the W3C's XML Encryption Syntax and Processing Candidate Recommendation [100]

to achieve mutual authentication and two-level confidentiality, while using the W3C's XML-DSIG standard [126] to maintain policy integrity and non-repudiation.

- ***Mutual Authentication Mechanisms***

Authentication provides means for one party in a transaction to determine the identity of the other party in the transaction. Because of issues with complexity, many existing applications are designed so they do not require client-side authentication. In other words, the authentication is conducted in one direction, mostly a client or user authenticating themselves to a server side.

In the distributed privacy protection model proposed by the author, it is highly desirable to require both parties involved in privacy interaction to authenticate each other, in order to prevent not only man-in-the-middle attack but also potential malicious attacks from both sides. It is suggested that the mutual authentication is implemented between PPAP and PPDP, between PPDP and PPEP, between PPDP and context-aware applications, as well as between context-aware applications and PPEP. In particular, it is important for a PPEP to authenticate the identity of the PPDP (i.e. Privacy Agent) from which it receives information disclosure decisions. Otherwise, there is a risk that an adversary could provide false or invalid authorization decisions, leading to a privacy violation. It is equally important for a PPDP to authenticate the identity of the PPEP and assess the level of trustfulness to determine if information disclosure decision could be passed to it.

Many different techniques may be used to provide two-way authentication, such as a private network, a VPN or digital signatures. In this work, the mutual authentication is conducted as part of SSL protocol that is based on Public Key Infrastructure (PKI) technology. The public key cryptography allows key exchange to happen over unsecured connections without compromising the security of the encryption process. (Note that using a point-to-point scheme like SSL may lead to other vulnerabilities when one of the end-points is compromised). Using SSL, every party in the distributed

privacy protection model is assigned a pair of private key and public key, one party authenticates itself to another through its public key certificate and by showing its knowledge of the corresponding private key. For instance, once the Privacy Agent encloses its information disclosure decision in a credential, that credential is signed by the Privacy Agent and includes a hash of the context-aware application's public key. PPEPs when presented with the credential can then verify that the credential indeed belongs to a information requester by hashing the public key corresponding to the private key in the SSL authentication, and then comparing that hash with the hash in the credential. If they match, then the PPEPs can be assured that the requester's public key matches the public key of the credential and that the requester is who originally hold this credential.

- ***Two-level Confidentiality Mechanism***

Confidentiality is required to ensure that information is accessible only to those authorized to have access. There are two levels at which confidentiality mechanisms can be applied in the privacy protection model: one is confidentiality at the communication level; the other is confidentiality at the message level. Keeping information secret at the communication level makes it difficult for an adversary to know what steps might be sufficient to obtain unauthorized access. It can be achieved by employing SSL protocols. Maintaining confidentiality at the message level involves the use of encryption mechanisms. The XML Encryption Syntax and Processing Candidate Recommendation [100] from W3C can be used to encrypt all or parts of data collecting policy exchanged in the proposed privacy protection model.

- ***Digital Signature Scheme for Integrity and Non-repudiation***

Integrity mechanisms are required to ensure that messages have not been altered (including inserting or deleting part of message) since they were originally created in PPAP. SSL protocols provide certain integrity protection. In addition to using SSL mechanisms, this work recommends the use of digital signature schemes, in particular, the W3C's XML-DSIG standard [126], to maintain integrity. Since privacy policies,

such as the Privacy Contract, are distributed between parties and might be acted on at a later time, and travel with protected information, it would be useful to sign the policies.

In addition to integrity, the digital signature scheme provides non-repudiation, a term referring to preventing a party from denying previous commitments or actions [88]. Non-repudiation is not a core security requirement of many systems, but in the proposed privacy protection model, all the parties, the PPAP, PPDP, PPEP and Context Providers, should be responsible for information and decisions that they contribute to privacy decision. The non-repudiation mechanism is useful for the distributed privacy protection model, as it is largely based on trust relationships between various parties.

4.5.2 Single Point of Attack

Even if secure communication between parties in the privacy protection model can be guaranteed, there is single point of attack that can compromise the distributed model. Points of vulnerability include the Privacy Agent that stores privacy preferences and makes information disclosure decision, the PPEPs that hold personal sensitive information and enforce information disclosure decision, the PPAPs where individual privacy preferences are edited and updated, and context-aware applications that make information requests. Compromise at these points may lead to privacy violation. This section focuses on discussing the single point of attack made on the Privacy Agent. Issues about maintaining the security of PPEP, PPAP, and context-aware applications are not discussed and left to decisions by their implementers.

The Privacy Agent can be compromised by attacks coming from inside, such as, the problem of bug, virus, worm, or spyware. Since the Privacy Agent contains three important repositories: the *Privacy Preferences Repository* that stores a person's privacy preferences, the *Resource Repository* which holds the information about context providers, and the *Context Information Repository* that stores a person's context

information, the threats caused by bugs, virus, worm, or spyware could at best disfunction the Privacy Agent, and in the worst case, lead to unauthorized disclosure of information stored in the Privacy Agent.

Attacks on the Privacy Agent may also come from outside, e.g. by message altering and by a malicious message replay attack, as discussed in the previous section. The message altering threat causes problem especially when the Privacy Agent does not develop appropriate exception-handling mechanisms to deal with malicious message forms. The message replay leads to a denial of service attack. For instance, an adversary or hijacked context-aware application simply replays its data requests and data collecting policies to the Privacy Agent even without having to understand the content of the request. Considering that the Privacy Agent may have a limited capability in terms of computing power and memory, a continuous influx of malicious repeated messages would soon slowdown the process of concurrent legitimate requests, and more severely lead to a complete shutdown of the Privacy Agent.

Mitigation Mechanisms

The single point of attack is certainly not unique to the privacy protection model and the privacy agent technology proposed in this work. One the one hand, the problem of bugs during design time, as well as the threat of spyware, virus and worms during run-time, are often encountered when developing and using software. The problem is generally hard to resolve in the domain of software development. Some possible solutions to mitigate the problem include: digital signature schemes to verify the source of the data (so that a user knows who to complain to or who to pursue legal recourse against), white hat hackers who check the behavior of code, and non-technical mechanisms such as magazine reviews [57]. On the other hand, combating denial of service (DOS) attack is a very difficult task. As shared by security community, the difficulty lies in mechanisms to determine what constitutes a legitimate session versus an illegitimate session especially when many attacking points belong to innocent people

with compromised systems [127]. The security discussion in this work does not contribute to addressing these threats. It relies on existing solutions and leaves to implementers to select appropriate mechanisms to mitigate problems in their system environment. Instead, appropriate exception-handling mechanisms are implemented in the Privacy Agent to react to the unpredictable nature of the dynamic context-aware environment. The mechanisms are there to ensure that the Privacy Agent will not render an incorrect authorization decision as a result of message modification attack.

4.5.3 Trust Concerns

Like many other distributed architectures, there are trust concerns haunting the vulnerability of the proposed privacy protection model in addition to security threats. If, ideally, transmission between parties in the privacy protection model would be guaranteed, every party would have good intentions, no adversary would attempt to deceive others by pretending who it is not, and hardware would never fail, then the question arises how much parties in the privacy protection model will trust the information and decisions of others upon which their decisions rely. Four trust relationships must be established to ensure the effectiveness of the proposed privacy protect model. They are discussed as following:

- First, people need to trust decisions made by their Privacy Agent. They need to believe that the Privacy Agent would (subject to the security threats discussed previously) be capable of making information disclosure decisions according to their preferences.
- Second, people and their Privacy Agent need to believe that PPEPs (i.e. holders of personal information) trust decisions made by Privacy Agent and would comply with the Privacy Agent when dealing with information access requests from context-aware applications.
- Third, the Privacy Agent needs to trust that Service Providers would comply with their stated data collecting policies as well as what the Privacy Agent agrees, once

they receive (from PPEPs) personal information.

- Fourth, the Privacy Agent needs to trust that Context Providers provide it with correct and updated context information when requested, so that it can make the appropriate information disclosure decision.

Trust Establishment and Enhancement Mechanisms

There are basically two types of trust involved in the privacy protection model. The first type is human trust, in particular, the trust between People and their Privacy Agent. The problem of how much people trust their privacy agent to make the "right" decision that complies with their expectation and need is difficult (if not impossible) to resolve. As discussed in the literature review work on computational trust (section 3.3.5), human trust is certainly a complex issue, and there is perhaps a fundamental incompatibility between our human notion of trust and the computational processes that try to mirror them.

In this work, the author does not incorporate computational trust mechanisms that aim to mimic human trust, but instead takes the following approach to enhance human trust of their Privacy Agent. Above all, appropriate privacy announcement mechanisms are advocated to enhance users' trust in disclosing their personal information by providing them with additional background information about information disclosure. Secondly, ontology-based modeling and reasoning techniques are exploited by the author to facilitate automated privacy control. It is proposed that these techniques make it possible for the Privacy Agent to interpret semantically-rich privacy preferences properly. Thirdly, various security safeguard mechanisms, as suggested in the previous section, are taken up to enhance the reliability of the Privacy Agent by mitigating potential security threats in the model. Finally, logging mechanisms are to be established in the Privacy Agent to keep all records of its decision. This makes it possible for people to audit the activities of their Privacy Agent and to ensure that the Privacy Agent is capable of handling sensitive information properly.

The second type of trust is machine trust, in particular, how the Privacy Agent selects and evaluates whether a service provider, context provider, or privacy policy enforcement point (PPEP) is trustful. Acquiring machine trust has been addressed in related work. For instance, Seng and Arbaugh [128] proposed a three-layer trust establishment model which provides a unifying view of trust establishment, consisting of authentication process, semantic representation of trust, and trust evaluation. Shand et al. [124] introduced a trust framework in which individuals compute their trust of information by combining their own trust assumptions with others' recommendations. Bertino et al. [129] presented a trust negotiation framework, which allows entities to establish mutual trust on first contact through an exchange of digital credentials. Other trust acquirement mechanisms include verifying a requester's reputation ranked by independent third parties [67] and checking "spam request" and black listings [15]. The author's work relies on these existing solutions for trust acquisition and establishment.

Another concern relevant to the machine trust is about how to ensure that Service Provider and PPEPs would comply with what they state and what the Privacy Agent agrees. Like Langheinrich [55] and Hong [57], the author recognizes the difficulty of providing a perfect privacy protection (if there is such a thing) by employing technical mechanisms alone. The proposed privacy protection model relies on social and legal pressures to compel parties to comply with stated privacy policies.

The author develops a Privacy Policy/Preference Language to facilitate the expression of privacy policy and preferences that comply with existing and emerging legislation in information protection and privacy, and provides appropriate mechanisms for communicating the privacy expression in a distributed privacy protection model. This will be described in detail in the next Chapter.

4.6 Chapter Summary

This chapter presented the author's work on the distributed protection model and intelligent agent technology to preserve individual privacy in context-aware ubiquitous computing environments. The work followed the guideline principles that were set forth in section 3.5, and strived to address limitations of existing solutions and tackle technical challenges that were identified in section 3.4. The component design of the Privacy Agent technology and reasoning behind the design has been discussed. The author also discussed the limitations of the proposed solution, mainly in terms of security threats and trust concerns, and suggested some possible safeguard and mitigation measures.

Three key discussions that are conducted in this Chapter and highlight the author's privacy solution are:

- *A new understanding of ubiquitous systems as a composite environment* that takes into account heterogeneous types of ubiquitous systems, including both constrained SmartSpaces and ubiquitous architectures that are based on specific network systems, such as the mobile network.
- *A distributed privacy protection model* that separates the privacy decision process from the enforcement process and thus allows multiple parties to participate in privacy protection with clear accountability of each party.
- *An intelligent agent technology* that facilitates automated processes of privacy control, and enables relatively unobtrusive user participation in controlling the disclosure of their sensitive information.

The author will present in Chapter 5 and Chapter 6 key technologies and novel approaches that facilitate the automated processes of the Privacy Agent, and demonstrate in Chapter 7 how the proposed privacy protection model and agent technology may be implemented in a real world scenario.

Chapter 5: Developing A Privacy Policy/Preference Language

To facilitate automated processes of the Privacy Agent, a Privacy Policy/Preference Language is developed so that various parties involved in the distributed privacy protection model and functional components of the Privacy Agent can have a common understanding of the necessary privacy requirements while interacting with each other. This chapter presents the author's work on developing semantic and syntactic specification of the Privacy Policy/Preference Language by adapting the W3C's Platform for Privacy Preferences Project (P3P) practices [95], and gives examples demonstrating how the language can be used by individuals to express their privacy preferences and by data collectors to state their data collecting policies. Reasoning behind major adaptation considerations will also be given. In the last section of this chapter, the author investigates previous ubiquitous computing work that attempted to apply the P3P practices to the ubiquitous computing environment as well as existing efforts to develop policy languages for access control and security policies. This investigation of related work indicates limitations in the approaches taken to date and help justify the author's effort to develop a new lightweight language for privacy expression toward a dynamic context-aware environment.

5.1 The Development of Privacy Policy/Preference Language

To facilitate automated processes of the Privacy Agent, the Privacy Policy/Preference Language is developed so that various parties involved in the distributed privacy protection model and functional components of the Privacy Agent can have a common understanding of the necessary privacy requirements while interacting with each other.

As its name suggests, the Privacy Policy/Preference Language provides vocabulary and syntax used in two ways —— for data collectors to state data collecting policy, and for individuals to express their privacy preferences. Some specific objectives that the privacy policy/preference language tries to achieve are:

- Being *generic*, in other words, to state application-independent data collecting policies, and to be user-adaptable in response to varieties of individual privacy preferences
- Being able to construct semantically-rich preferences in a *standardized, machine-readable format* that can be retrieved automatically and easily interpreted by the Privacy Agent.
- Being *descriptive* to facilitate human understanding.
- Being able to express *context-dependable* privacy preferences towards dynamic context-aware ubiquitous computing environment.
- Being *lightweight* in response to the requirement of managing privacy in resource constrained environments.

The Privacy Policy/Preference Language adapts terminology and policies specified in the W3C's Platform for Privacy Preferences Project (P3P) [95], and has taken into consideration preference expressions defined in P3P preference formulation languages such as APPEL [105] and Xpref [106]. The W3C's P3P specification is probably the most significant effort to enable web users to gain control over their private information [106]. The designers of P3P simultaneously designed a preference language called APPEL to allow users to express their privacy preferences, thus enabling automatic matching of privacy preferences against P3P policies. Since significant work has gone into making the P3P specification comply with existing and emerging legislation in information protection and privacy, by grounding the development of the Privacy Policy/Preference Language on the P3P practice, the author expects to benefit from the substantial legal and social expertise that has been put into the development of the P3P

standards.

Since the P3P practice is an attempt to provide privacy mechanisms for the Web, its base data schema and policies, as well as the P3P-based preference formulation languages (such as APPEL[105] and Xpref[106]), do not take into account some types of information and some types of interactions presented in context-aware ubiquitous computing environments, changes are necessary to the P3P standards before they could be useful in protecting individual privacy in context-aware ubiquitous computing environments. The approach taken by the author to adapt the P3P specification does not use the extension mechanism provided by P3P syntax to make direct add-on to the P3P specification, but instead borrows relevant concepts and definitions from the P3P standards to construct a new lightweight language tailored to the requirement of managing privacy in resource constrained environments. The following sections describe in detail the author's effort to develop base data schema, policy and preferences elements of the Privacy Policy/Preference Language and to use them to construct data collecting policies, privacy preferences and privacy agreements respectively.

5.1.1 Developing Base data schema

The P3P specification predefines a base data scheme, consisting of data that can be referenced in accord with P3P policies. This allows semantic agreement on data collected, so that there is no ambiguity between a user and a website about what is exactly collected. Following the P3P practice, the author defines a base data schema of the proposed Privacy Policy/Preferences Language, to contain personal information that is deemed sensitive and required to be protected in a context-aware ubiquitous computing environment.

Since the P3P is an attempt to provide privacy mechanisms for Web browsing and online transactions, its base data scheme only takes into account a person's identifying

information (such as name, birthday, home-address, credit card details, etc.) as private data to be protected. In context-aware environments, dynamic contextual information (such as a user's current location and activities) is also sensitive, but is not covered by the P3P specification. Consequently, additional data elements need to be defined to account for an enlarged scope of sensitive information, and to include dynamic (i.e. changeable) personal information. In addition, the Privacy Policy/Preference Language is used for individuals to specify privacy preferences. Additional data elements that should be added into the base data schema include contextual conditions that are used by individuals in specifying individual privacy requirements. For instance, a user might specify that her colleagues are allowed to know her location only when she is working, or when she is in holiday, when she is in the town. Here, the "colleagues" and "working" are such contextual condition. The author categorizes three primary types of contextual knowledge as disclosure condition: Location, Time, and Human Activity, and include them in the base data schema.

Table 5.1 presents some staple information that consists of the base data scheme of the proposed Privacy Policy/Preference Language. It indeed incorporates some user data defined in the P3P specification, and includes temporal-spatial contextual information of human activity. As illustrated in Table 5.1, the information is categorized to facilitate individuals to express generalized preferences and rules over the exchange of their data. This approach of grouping information also provides hints to individuals and their Privacy Agent regarding information sensitivity. Note that Table 5.1 shows only one facet of the categorization of information. Some information from different categories could be extracted to form new categories according to the users' own preference. One such important category is *identity*. In this work, the *identity* is referred as information that can be reasonably tied to an individual. Defining the scope of identity information is not straightforward and depends on various perceptions and purposes. It includes social recognition like a real name and student ID, but also could be an IP address, a Messenger account and/or an e-mail address if it is felt that they can

be tied to and distinguish a particular individual. A pseudonym is also important identity information that helps deliver personalized services while preserving privacy.

Table 5.1 *Staple information in the Base Data Scheme*

	Category	Context Information
Personal Information (static)	Profile	gender, age, birth date, etc
	Identity	real name, pseudonym, anonymous, etc
	Contact	business contact (e.g. company website, business email, office number, etc) home contact (e.g. home address, home phone, etc) school contact (e.g. school name, school address, etc)
	Socialgroup	Family member, alumni, colleagues, a group of people that a person knows, etc.
	Online	homepage, instant messaging, chat ID, etc.
	Finance	bank account, credit card, income, etc.
	Profession	employer, job title, occupation, membership, etc.
Contextual Knowledge (dynamic)	Location	a person's current location, etc.
	Time	current time, etc.
	Activity	working, traveling, meeting, etc.

The author has followed a formal ontology-based method to represent the categorization of the base data schema. As a result, the privacy-sensitive personal information is modeled in a hierarchical way, and the ontology-based description logic can be used to reason over hierarchical structuring of the information as a means of facilitating semantic matching. This makes it possible to control information disclosure to various levels of accuracy (i.e. ambiguous disclosure). A concrete example is that users could choose to disclose their location information at city, district, or street level. The work on the ontology engineering will be presented in the next chapter, in particular, section 6.2 Privacy Preference Rule Ontology, and section 6.3 Context Information Modeling.

5.1.2 Developing Policy Elements

In the author's proposal, interactions between a human user and a context-aware application are initiated by the application sending a data request with associated data

collecting policies to the user's Privacy Agent. Like P3P policies, the data collecting policies are meant to enhance users' trust in disclosing their personal information by providing them additional background information about the disclosure. Figure 5.1 exemplifies a data collecting policy of the *Who-near-me* service (recalling the use scenario described in section 4.1) that asks for individual real identity, social contact and location information, in order to notify him when his friends or family member are nearby.

```
<POLICIES>
<POLICY>
  <ENTITY>
      <APPLICATION> Who-near-me </APPLICATION>
      <ID>0101 </ID>
      <SERVICE-PROVIDER> O2 Mobile Operator</SERVICE-PROVIDER>
      <CATEGORY> Tourism </CATEGORY>
      <REQUST-MODE> continuous </ REQUST-MODE>
      <SERVICE-MODE> optional </ SERVICE-MODE>
  </ENTITY>
  <STATEMENT>
      <CONSEQUCNE> We try to help you find if any of your friends, colleagues, alumni are
                   in the city. We ask for your location information and social contact
                   to make the service available, and will retain them until you opt out
                   of the service. </CONSEQUCNE>
      <PURPOSE> navigation </PURPOSE>
      <RECIPIENT> ours </RECIPIENT>
      <RETENTION> stated-purpose </RETENTION>
      <DATA-GROUP>
        <DATA ref="# personalinfo.identity.realname">
        <OPTION>
          <TARGET-DATA ref="#personalinfo.socialgroup.friend">
          <TARGET-DATA ref="# personalinfo.socialgroup.colleague">
          <TARGET-DATA ref="# personalinfo.socialgroup.alumni">
        </OPTION>
        <OPTION>
          <DATA ref="# personalinfo.location.city">
          <DATA ref="# personalinfo.location.city.district">
          <DATA ref="# personalinfo.location.city.district.street">
        </OPTION>
      </DATA-GROUP>
  </STATEMENT>
</POLICY>
</POLICIES>
```

Figure 5.1 *An example of data collecting policy*

As illustrated in Figure 5.1, a data collecting policy consists of a set of policy elements that regulate the processing of personal data. The following parts describe major elements to construct a data collecting policy. The author refers these elements as *Policy Elements*, which in contrast to *Preference Elements* that are used to express privacy preferences and will be presented in the next section.

A data collecting policy is enclosed in the <POLICIES> element that allows a single file to hold multiple policies, each being uniquely identified by an ID attribute and corresponding to one application. The <POLICY> consists of two blocks, providing information about data collector (<ENTITY>) and data collected (<STATEMENT>) respectively. Different from the P3P specification, where the <ENTITY> block provides a mechanism for describing business and contact details of an organization offering Web-based services, entities in the author's proposal represent context-aware applications that issue data collecting requests and policies.

The <ENTITY> block contains information about data collectors in terms of application name (<APPLICATION-NAME>), ID number (<ID>), service provider (<SERVICE –PROVIDER>), category (<CATEGORY>), request mode (<REQUEST-MODE>) and service mode (<SERVICE-MODE >). The ID number is used to uniquely identify an application, while indicating service provider information (the <SERVICE-PROVIDER> element) enables privacy protection solutions to introduce trust-based mechanisms, as people and their privacy agent may have specific views about which service providers to trust and to disclose information (for example, only to those certified by trusted organizations or authorities). The <CATEGORY> element is used to filter applications within the same genre. The P3P specification gives some examples of application categories, while in the author's proposal, service providers can also set attribute values of the <CATEGORY> element according to their own needs. The <CATEGORY> makes it possible to place disclosure restrictions on certain types of applications, so that people can express more generalized preferences and rules over the exchange of their data. For instance, a person can place information disclosure restrictions on all tourism applications.

The <REQUEST-MODEL > and <SERVICE-MODEL> elements are defined to correspond to the type of interactions presented in context-aware ubiquitous computing environments and application behavior therein. First, unlike request-reply web

interactions, information exchanges in context-aware ubiquitous computing environments may not be one-off; quite often, they continue for a certain period of time. For example, a location-tracking application asks for a user's location over a period of time. The <REQUEST-MODE> element is specified to reflect such a requirement. As exemplified in Figure 5.1, the *Who-near-me* application states that it will collect user data continuously. Second, since the information collecting in context-aware ubiquitous computing environment may be continuous, the service mode of context-aware applications can be optional or mandatory. As depicted in privacy protection scenarios in section 4.1, the use of *Who-near-me* service is optional and subscribed users can choose to opt out from the service later on whenever they want, while the *Emergency Notification* service is mandatory, in that mandatory data collecting is continuously running and cannot be deactivated by a user once users enter a service agreement with it. This is very similar to the application of security cameras in a supermarket. The mandatory data collecting is required in crisis situations and for safety concerns. As identified in the section 3.2, people have concerns about overriding their privacy needs for the sake of safety.

Like the P3P policies, the <STATEMENT> block in the author's proposal is specified to give a detailed account of personal data collected by applications, as well as background information about the data collecting. The background information includes a human-readable description of the effects of data collecting (<CONSEQUENCE>), potential recipients of data (<RECIPIENT>), purposes of data collecting (<PURPOSE>), and duration of intended use (<RETENTION>)[11]. The author retains attribute values of the <RECIPIENT> and <RETENTION> elements defined in the P3P specification, and limits that only one attribute value of <RECIPIENT> and <RETENTION> element is specified in a data colleting policy. In addition, the author replaces P3P's <PURPOSE> attributes with specification tabled in Table 5.2, to cover a wide variety of purposes that are used in the context-aware ubiquitous environments.

[11] The usage of these elements and their attribute values can be found in the P3P specification [95].

The reason of such a replacement is as follows.

Table 5.2 *Purpose Declaration*

Attributes	Description
Navigation	Personal information (in particular, location data) is collected to guide user to either a specific point of interest (e.g. nearby bank), or to provide additional information about the current location (e.g. tour guide).
Marketing	Personal data (such as gender, purpose history, etc) is collected to provide personalized service and information to users.
Multimedia	Information is collected in order to provision multimedia applications, e.g. live video streaming or audio recording.
Communication	Information is collected in order to provide appropriate communication network capabilities to users.
Health	Personal data is collected to provide health advice.
Finance	Personal data is collected to provide financial advice.
Social_analysis	Personal data is collected for social analysis (e.g. demographic statistics). The analysis might or might not need a person's real identify.
Develop	Personal data is collected for research and development purpose of the service providers that collect the data.
Security	Personal data is collected for security reasons and legal surveillance.

The P3P specification defines some types of purposes to indicate why a data collector requests certain information. However, the P3P's purpose element reflects its orientation to e-commerce and Web interactions. It defines information usage primarily relevant to Web services and uses *Current* purpose as a versatile declaration to cover more uncategorized information usages when users are engaging in certain service, e.g. to return search results, give access to an online address book, or renew a subscription. Such an approach is feasible in Web environments because explicit interactions exist between users and websites. Clicking on hyperlinks or buttons and filling out form fields either manually or semi-automatically using a built-in browser imply a user's acknowledgement of transactions. In ubiquitous computing environments, however, interactions are prone to happening unobtrusively, typically without an explicit

involvement, even without acknowledgement, of users. As a result, information usages that might prevail in context-aware ubiquitous computing environments cannot be addressed by using the "current purpose" declaration. For this reason, the author retains the good practice of using the PURPOSE element but replacing its attributes to cover a wide variety of purposes that are used in the context-aware ubiquitous environments.
Note that the intended use of the PURPOSE declaration is different from that of the <CATEGORY> statement in <Entity> block. The <Category> information is specified by service providers to reflect the function of context-aware applications. It is defined in an arbitrary manner and its use is optional. In contrast, attributes of <PURPOSE> element are predefined in the proposed Privacy Policy/Preference Language. In other words, they are intended to be shared by all involving parties (e.g. Privacy Agent, Service Provider, and context-aware applications) in privacy interactions.

Like the P3P policies, the <STATEMENT> block in the author's proposal includes a <DATA-GROUP> element, stating the data that applications are about to collect. It is likely that context-aware applications will offer optional information granularity levels for user disclosure. The author thus introduces a new <OPTION> element in <DATA-GROUP> sub-block, to specify that at least one of the items in the <OPTION> element should be chosen. The data contained in <DATA-GROUP> sub-block are differentiated and tagged with <TARGET-DATA>, which points to social group information to which personal information is disclosed. The <TARGET-DATA> element is different from the <RECIPIENT> element that represents potential recipients of data, in that data tagged with <TARGET-DATA> are themselves sensitive personal information and will be disclosed. For instance, in the *Who-near-me* service, a person's location information is required to be disclosed to his/her alumni, colleagues, or friends. Here, the alumni, colleagues, and friends are all target data.

To sum up, Figure 5.2 presents a high-level skeleton of policy elements arrangement in data collecting policies. It highlights policy elements introduced by the author in red, while marking terms and concepts that are borrowed from the P3P specification and

with some change in green. The black elements are adopted directly from the P3P specification without any changes. A full syntax and semantic of policy elements defined in XML schema is presented in Appendix D.

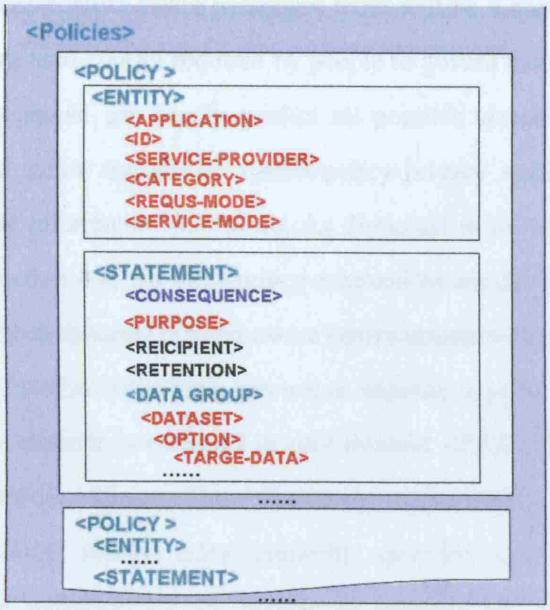

Figure 5.2 *A high level skeleton of the arrangement of policy elements*

5.1.3 Developing Preference Elements

The previous section described the policy elements that are used by data collectors to state their data collecting policies. Another part of the Privacy Policy/Preference Language is syntactic and semantic constructs that are used by individuals to express their semantically-rich privacy requirements. The author refers to the constructs as *Preference Elements.* Unlike the development of policy elements, which is based primarily on adapting P3P policies to reflect interactions presented in context-aware environment, the author's work on developing the syntax and semantics of preference elements does not rely on the specification of the P3P-based preference languages, in particular, APPEL. The author's study with P3P and APPEL identified that APPEL and approaches based on extending APPEL are insufficient as a preference language to express semantically-rich privacy preferences. The limitation is associated with

fundamental problems in the design of APPEL, and cannot be overcome without a complete redesign of the language[12].

It is likely that, in the context-aware paradigm, there will be a considerable variety of rules and levels of desired control required by people to govern their perceptual context and privacy. Since people can hardly predict all possible scenarios for granting or denying access, the author introduced a meta-policy privacy mechanism to facilitate high-level control of information disclosure. As discussed in the working logic of the Privacy Agent in section 4.4, the meta-policy mechanism would enable individuals to manage their privacy in dynamic context-aware environments with relative ease. Figure 5.3 illustrates a syntax that individuals can use to stipulate a privacy meta-policy. The <META-POLICY> element is enclosed in root element <PREFERENCES>, and has two predefined attribute values. The "*Optimistic*" value means that by default all requests are permitted, except rules explicitly specified by users, whereas the "*Pessimistic*" value indicates that by default all requests are denied, except rules explicitly specified by users.

```
<PREFERENCES>
<META-POLICY> optimistic </META-POLICY>
</PREFERENCES>

<PREFERENCES>
<META-POLICY> pessimistic </META-POLICY>
</PREFERENCES>
```

Figure 5.3 *Specifying privacy meta-policy*

In addition to having a high-level coarse control of information disclosure through the meta-policy, individuals can use the Privacy Policy/Preference Language to construct more specific preference rules, to limit information disclosure with respect to data collecting practices and/or in response to certain contexts. The following parts describe major elements to construct privacy preference rules.

When individuals specify their privacy preferences, all the preferences are enclosed in

[12] The author will discuss more about this point in related work in section 5.2.

the <PREFERENCES> element that allows a single file to contain multiple preference rules, in addition to the specification of the meta-policy. Every preference rule is expressed with three elements: Rule behavior (<BEHAVIOR>), Data (<DATA>), and Condition (<CONDITION>).

Every preference rule must specify one and only one <BEHAVIOR> element with one of the attribute values: "Permit", "Forbid" or "Askme". The "Askme" attribute is specified when explicit consent is required by a person for disclosing his/her personal information. As exemplified in the privacy protection scenarios in section 4.1, Alice specifies that any services or applications requiring her real identity and exact location must have her explicit consent.

The <DATA> element refers to sensitive personal information which is protected. Its attributes are those defined in the base data scheme specified in section 5.1.1. Multiple data elements may be enclosed in a preference rule, and a rule with empty data implies that its rule behavior is applicable to all data under the conditions specified.

Each preference rule must specify conditions under which information is or is not to be disclosed. Two types of conditions exist: Policy Condition and Context Condition. The policy condition constrains data collecting practices to respond to their Purpose, Recipient and Retention. The attribute values of the policy condition <Purpose>, <Recipient> and <Retention> are identical to that of correspondent policy elements defined in the previous section. The context condition is constrained to limit information disclosure to respond to specific contexts (Target, Location, Time and Activity). As their names suggest, <Target> element indicates to whom information should or should not be disclosed, it is the counterpart of the <TARGET-DATA> policy element used in data collecting policy. The <Location>, <Activity> and <Time> elements limit information disclosure when a person is in certain location, engaged in some activities or at a specific time. Attribute values of these elements are defined in the base data schema.

The following parts give three examples, demonstrating how the *Preference Elements* are used to construct various privacy preferences, which are not only to limit information disclosure with respect to data collecting practices, but also in response to certain contexts.

Preference 1. *Applications can use my pseudonyms, personal profiles and location to deliver personalized services without alerting me, as long as the information is not use for "develop" purpose. However, any applications requiring my real identity and exact location must have my explicit consent.*

```
<PREFERENCES>
<RULE behavior="permit"
  data="personalinfo.identity.pseudonyms"
  data="personalinfo.profile.*"
  data="personalinfo.location.*"
  condition="/POLICY/STATEMENT/PURPOSE/*
              [value(.) != "develop" ]"
  condition="/CONTEXT/* []" />
<RULE behavior="askme"
  data="personalinfo.identity.realname"
  data="personalinfo.location.city.district.street"
  condition="/POLICY/* []"
  condition="/CONTEXT/* []" />
</PREFERENCES>
```

Figure 5.4 *An example of privacy preference*

The preference 1 exemplifies a restriction on data collecting practice (i.e. "not for *'develop'* purpose"). The data to which disclosure conditions apply are stated before the corresponding conditions. Corresponding to the data collecting policies exemplified in Figure 5.1, a rule's condition starts from its root node and descends to the POLICY node, and then further descends to the STATEMENT node and PURPOSE node. The "*" operator selects all child elements of the PURPOSE node. Similarly, "personalinfo.location.*" refers to the location information at any levels of granularity, while "personalinfo.profile.*" includes all sorts of personal profile information, such as gender, age, etc. The author specifies three different rule behaviors: "permit", "forbid", and "askme".

Preference 2. *My colleagues are allowed to know my location only when I am working, while my family members and close friends can know my location when I am not working.*

```
<PREFERENCES>
<RULE behavior="permit"
  data=" personalinfo.location.*"
  condition="/POLICY/[]*
  condition="/CONTEXT/[
              TARGET/*[value(.)="personalinfo.socialgroup.colleague" ]"
          and ACTIVITY/*[value(.)="personalinfo.activity.working"]
          ] " />
<RULE behavior="permit"
  data="personalinfo.location.* "
  condition="/POLICY/[]*
  condition="/CONTEXT/[
              TARGET/*[value(.)="personalinfo.socialgroup.family" or
                       value(.)="personalinfo.socialgroup.friend" ]
          and ACTIVITY/*[value(.) !="personalinfo.activity.working"]
          ]" />
 </PREFERENCES>
```

Figure 5.5 *An example of privacy preference*

The *Preference 2* takes context information (i.e. working) as a disclosure constraint. The author's specifies a condition element (i.e. "condition = "/context/*") to respond to users' privacy requirement of that context. Once the *Policy Evaluator* in the Privacy Agent detects such a condition, it will make the context check.

Preference 3. *My location information (at all levels) should not be disclosed to my colleagues, friend, and alumni for the policy condition declared in the who-near-me service once I leave London city.*

```
<PREFERENCES>
<RULE behavior="forbid"
  data=" personalinfo.location.*"
  condition="/POLICY/STATEMENT/
            PURPOSE/* [value (.)="navigation"]
        and RECIPIENT/* [value(.)= "ours"]
        and RETENTION/* [value(.)="stated-purpose"]"
  condition="/CONTEXT/[
            TARGET/* [value(.)="personalinfo.socialgroup.colleague" or
                      value(.)= "personalinfo.socialgroup.friend" or
                      value(.)= "personalinfo.socialgroup.alumni"]
        and LOCATION/* [value(.)!="london"]
            ]" />
</PREFERENCES>
```

Figure 5.6 *An example of privacy preference*

The *Preference 3* demonstrates a more interesting case which combines context constrains and policy conditions (i.e. conditions pointing to data collecting policies) in a single preference rule. Based on evaluating these user-specified privacy preferences, the Privacy Agent will be able to judge the acceptability of the data collecting policies of applications, and make appropriate decisions on information disclosure. Unlike P3P-based agents, which compare data collecting policies and preferences based on a simple syntax matching, the author introduces context-aware intelligence into privacy rule evaluation processes and attempt to take advantage of ontology-based description logic to reason over the semantic relationship between information. This gives individuals a great flexibility to express their privacy requirements at various levels of abstraction.

Note that the preference rules presented in this section are illustrated using Xpath language [89]. The Xpath is only used for presentation purpose. Approaches taken by the author to conduct privacy policy and preference evaluation are not based on pure syntax of Xpath, but instead, employ ontology-based techniques to conduct semantic reasoning. This will be discussed in the next chapter.

5.1.4 Privacy Contract

In addition to constructing data collecting policies and privacy preferences, the privacy vocabulary and elements specified in previous sections are also used to describe the *Privacy Contract.* As discussed in the section 4.3, the *Privacy Contract* is a disclosure agreement generated by the Privacy Agent once a positive information disclosure decision is made. Intended data collectors render the *Privacy Contract* obtained from the Privacy Agent to the corresponding PPEPs for information access.

The layout of a privacy contract file is not much different from that of data collecting policies files. The Privacy Agent generates a privacy contract by modifying the data collecting policy file according to the users' privacy preference. It mainly changes the data contents enclosed in the <DATA GROUP> element of the data collecting policies while retaining what it agrees. Figure 5.7 below exemplifies a privacy contract file regarding the *Who-near-me* application. According to the privacy protection scenarios depicted in section 4.1, Alice instructs her Privacy Agent that she would like to accept the *Who-near-me* service offer and accepts the compromise of her wish for privacy, i.e. disclosing her location information to her friends, colleagues and alumni, but only at city level. As illustrated in the Figure 5.7, a privacy contract is enclosed in <PRIVACY CONTRACTS>element that allows a single file to hold multiple contracts, each corresponding to one service offer.

```
<PRIVACY CONTRACTS>
<PRIVACY CONTRACT>
<ENTITY>
    <APPLICATION> Who-near-me </APPLICATION>
    <ID>0101 </ID>
    <SERVICE-PROVIDER> London Tourist Center </SERVICE-PROVIDER>
    <CATEGORY> Tourism </CATEGORY>
    <REQUST-MODE> continuous </ REQUST-MODE >
    <SERVICE-MODE> optional </SERVICE-MODE >
</ENTITY>
<STATEMENT>
    <CONSEQUCNE> We try to help you find if any of your friends, colleagues, alumni are
                 in the city. We ask for your location information and social contact
                 to make the service available, and will retain them until you opt out
                 of the service. </CONSEQUCNE>
    <PURPOSE> current </PURPOSE>
    <RECIPIENT> ours </RECIPIENT>
    <RETENTION> stated-purpose </RETENTION>s
    <DATA-GROUP>
      <DATA ref="#personalinfo.identity.name">
      <OPTION>
        <TARGET-DATA ref="#personalinfo.socialgroup.friend">
        <TARGET-DATA ref="#personalinfo.socialgroup.colleague">
        <TARGET-DATA ref="#personalinfo.socialgroup.alumni">
      </OPTION>
      <OPTION>
        <DATA ref="#personalinfo.location.city">
      </OPTION>
    </DATA-GROUP>
  </STATEMENT>
</PRIVACY CONTRACT>
</PRIVACY CONTRACTS>
```

Figure 5.7 *A privacy contract file regarding the data collecting policy of the Who-near-me service*

In addition, the author introduces a new <DATASET> element in the privacy contract construct to enable more complex agreement with respect to disclosure of data under different conditions. For instance, Alice could specify in her preferences that her friends and alumni could know her exact location, while her colleagues only get notified that she is in the city. As illustrated in Figure 5.8 below, the <DATA> elements and their associated <TARGET-DATA> elements are grouped in the same <DATASET>.

```
<DATA-GROUP>
  <DATA ref="#personalinfo.identity.realname">
  <DATASET>
    <OPTION>
      <TARGET-DATA ref="#personalinfo.socialgroup.friend">
      <TARGET-DATA ref="#personalinfo.socialgroup.alumni">
    </OPTION>
    <OPTION>
      <DATA ref="#personalinfo.location.city.district">
    </OPTION>
  </DATASET>
  <DATASET
    <OPTION>
      <TARGET-DATA ref="#personalinfo.socialgroup.colleague">
    </OPTION>
    <OPTION>
      <DATA ref="#personalinfo.location.city">
    </OPTION>
  </DATASET>
</DATA-GROUP>
```

Figure 5.8 *An example of <DATA-GROUP> data arrangement using the <DATASET> element*

5.2 Related work

- ***Previous work attempting to adapt the P3P practice in ubiquitous computing environments***

Applying P3P practices to ubiquitous computing environments has been previously proposed by few research groups [15, 55, 94]. In particular, Myles el al. [15] at University of Arizona and Lancaster University presented the first attempt to adapt P3P practice in a ubiquitous computing system. In the location-aware system proposed by them, a simple modification of the P3P policy was conducted to express privacy policies over the release of personal location information. The modification work mainly addresses the requirements of location-based applications and includes: 1) introducing the elements in the Entity block to differentiate different organization types (such as nonprofit, profit, or government) and to let the system introduce certification schemes, 2) developing a new set of broad classifications of Purpose attributes to more accurately reflect the intentions of location-based service providers, and 3) introducing

request-initiation method to differentiate unsolicited interactions that are not explicitly or consciously triggered by users (e.g. initiate speculatively as a user wanders into a particular region) from solicited interactions that have been explicitly triggered by some user-initiated action (e.g. requesting a taxi to a person's current location). The Privacy Awareness System (PawS) [55] developed by Langheinrich for the ubiquitous computing environment presented the most informative work that extended P3P policies to account for data collecting of sensor-based location information and through perception mechanisms such as cameras and microphones. It also provides methods to allow the query, update and deletion of data once they are collected. The Web-Architecture for Service Platform (WASP) project developed at University of Twente by M. Zuidweg [94] also used the P3P policy as a basis for privacy control in their context-aware service platform. They proposed to include contextual data, such as location, data, and users status (e.g. busy, on the phone, do not disturb) in the data schema of the P3P specification, and to differentiate single collection service interaction from tracking services that can automatically re-collect contextual data to trigger certain behavior upon change of context and without user interaction.

These previous efforts provide insights to tackle some difficulties in applying the P3P practice to the ubiquitous computing environment. The development of the syntax and semantics of the Privacy Policy/Preference Language here has built upon their experience. However, these earlier attempts have limited their P3P practices to only be a vehicle for data collectors to state their collecting requirements; appropriate mechanisms to facilitate individuals to express their privacy preferences were either limited (e.g. [94]) or not taken into account (e.g. [15, 55]). More specifically, following the P3P practice, PawS [55] and Myles's [15] did not develop privacy preference mechanisms. They instead suggested employing the P3P-based preference formulation language APPEL [105] to allow users to express their privacy preferences. The developer of WASP privacy architecture [94] proposed an XML-based language to express context-dependent preferences. The language made a simple extension of the APPEL to support constrains on the following types of context information as a privacy

preference (Location, Time, Data, Day of the week, and User Status). However, the author's study with the P3P and the APPEL has identified that the APPEL and APPEL-based methods are insufficient as a preference language to express semantically-rich privacy preferences towards dynamic context-aware environment. A thorough review of APPEL, conducted by IBM Almaden Research Center [106], has summarized major problems with the design of APPEL and suggested that the problems cannot be solved without a complete redesign of the language. The researchers at IBM instead proposed a new alternative language (called Xpref) to overcome the limitations of APPEL.

The Privacy Policy/Preference Language developed in this work provides an integrated solution for data collectors to state data collecting policies (in a manner that is adequately understandable by the Privacy Agent and users), and for individuals to construct semantically rich personal privacy preferences (in a standardized, machine-readable format that can be retrieved automatically and interpreted easily by the Privacy Agent). To some extent, the language can be seen as a combined vocabulary of the P3P policy elements and the Xpref preference constructs. However, the language developed by the author can be used to construct privacy preferences to limit information disclosure not only with respect to data collecting policies, but also in response to dynamic contexts. This is not supported by the P3P and the P3P-based preference formulation languages [105, 106].

More importantly, there is a fundamental difference when using the languages to perform privacy policy and preference evaluation. When using the Xpref and APPEL, the evaluation mechanism is based on a simple syntax matching of privacy preferences against the P3P policies. On the contrary, the approach taken by the author to conduct privacy policy and preference evaluation is not based on pure syntax of the language, but instead, employs ontology-based techniques to conduct semantic reasoning. Employing ontology-based methods to perform semantic policy analysis of individual privacy preferences is a key proposal of the author's work, and will be discussed in the

next Chapter.

- ***Other policy languages for security and privacy***

The use of rule-based policy is common in computing systems that feature security or privacy protection [143]. The security community has developed powerful policy languages to capture access control privileges. For instance, the XML Access Control Markup Language (XACML) [85], developed by the Security Services Technical Committee of OASIS, is a declarative access control policy language implemented in XML for distributed systems, the Security Assertion Markup Language (SAML) [130], also proposed by the OASIS, is an XML-based framework for communicating user authentication, entitlement, and attribute information, and the Enterprise Privacy Authorization Language (EPAL) [96], developed by IBM, is used for expressing access rights to information in an enterprise environment.

However, there are differences between using policy for security access control and for privacy protection. Privacy policy regulates the processing of personal data, and can be influenced by many factors, including laws, self-regulatory guidelines, ethics, and user preferences [62]. When using policy for privacy protection, the task of policy definition no longer rests on the shoulders of a system administrator, as users have varying privacy protection preferences and it is difficult to specify a single global policy that would suit the preferences of all users. A major feature that a privacy policy language should offer, to cope with dynamic context-aware environment, is to support personal privacy preferences in response to various contexts (e.g. Target, Location, Time and Activity). Although existing policy languages like EPAL, as its name suggests, also claimed privacy protection, they are indeed efforts made for facilitating access control for information protected in an enterprise environment. Unlike the Privacy Policy/Preference Language developed in this work, they can not be manipulated by people to express their individual privacy requirements.

5.3 Chapter Summary

This chapter presented the author's work on developing semantic and syntactic specification of the Privacy Policy/Preference Language by adapting the W3C's P3P practices. Reasoning behind major adaptation considerations of the P3P practices has been discussed. It also gave examples to demonstrate how the language can be used by individuals to express their privacy preferences and by data collectors to state their data collecting policies. The investigation of the related work in the last section of this chapter indicated limitations in approaches taken to date and helped justify the author's effort to develop a new lightweight language for privacy expression toward dynamic context-aware environment.

The next chapter will present how the author exploits ontology-based methods to represent the Privacy Policy/Preference Language developed in this chapter, and uses hybrid reasoning mechanisms to conduct semantic policy analysis of privacy preferences and context reasoning. They are key technologies to facilitate automated processing of the Privacy Agent, and to empower individuals to manage their privacy towards dynamic context-aware ubiquitous computing environments with relative ease.

Chapter 6: Ontology-based Modeling and Reasoning of the Privacy Policy/Preference Language and Context Information

This chapter presents the author's work on exploiting ontology-based methods to model the Privacy Policy/Preference Language developed in Chapter 5, and to use hybrid reasoning mechanisms for semantic policy analysis and context reasoning. They are key technologies to facilitate the automated process of the Privacy Agent, and to empower individuals to manage their privacy towards dynamic context-aware ubiquitous computing environment with relative ease. The chapter sets out by explaining reasons and objectives for using ontology technologies in this privacy protection work, as well as choices of ontology language used. Much work then goes into a detailed description of the author's approaches to use ontology-based technologies for policy representation and reasoning in addition to context modeling and reasoning. The final part of this chapter looks into emerging approaches that employ Semantic Web technology and ontology-based techniques for policy representation and reasoning, in addition to related work on context information modeling and reasoning. The investigation of related work indicates the novelty of the author's work and the advantages of approaches taken by the author.

6.1 Reasons and Specific Objectives for Using Ontology-based Methods and Choices of Ontology Language

In the literature review in Chapter 2, the author has compared different information representation and modeling techniques that are used in context-aware ubiquitous paradigm, and highlighted advantages of using ontology-based modeling approach in

terms of formalization, promoting reuse, and interoperability. In addition to these advantages stated in the literature, this section explains reasons and specific objectives for using ontology technologies in the author's privacy protection work, and discusses choices of ontology language used.

6.1.1 Reasons and Objectives of Using Ontology-Based Methods

There are basically three reasons why ontology-based approaches are particularly useful and should be used in the author's work to preserve individual privacy.

Firstly, it is likely that in context-aware paradigm there will be considerable variation in individual privacy preferences. Privacy preferences have semantic meanings, which may require policy analysis not only based on syntax, but also on meanings of policy. Ontologies are adept at encoding concepts and capturing relationships between concepts. The information relationships can be exploited by reasoning mechanisms based on ontology-based description logic to facilitate semantic analysis. Examples of two important information relationships that are particularly useful to policy evaluation work are as follows:

- **Subsumption:** For example, knowing that a person does not want disclose her home contact and her home address is a subset of her home contact information, the Privacy Agent could reason that it should also keep the person's home address secret.
- **Association:** For instance, knowing that a person does not want to reveal her home address and that her home telephone number is associated with her home address (e.g. under the same category), the Privacy Agent could reason that it should also keep the person's home phone number secret.

These information relationships are inherent in an ontology description, but may not be easy (if not impossible) to interpret by traditional logic programming methods based purely on syntax manipulation. The ontology-based semantic reasoning capability

plays an important role during privacy interactions in the proposed distributed privacy protection model. It facilitates privacy negotiation and policy analysis by providing semantic matching in the case that requested data and conditions, specified in data collecting policies, do not exactly match individual privacy preferences. The ontology-based methods offer semantically rich policy representation and simplify policy analysis.

Secondly, different social and legal environments lead to different levels of expectation from privacy protection, and users will need varying levels of desired control. This requires a shared model for expressing privacy policies and preferences between users and the environment and among users in the environment. To facilitate an automated process of privacy control, an important issue is to construct semantically rich personal privacy preferences in a manner that is understandable by users as well as easily transformable to the machine language. As identified in section 2.3, ontology-based models are capable of describing contextual facts and interrelationships in a precise and unambiguous manner, and thus allow participating parties in heterogeneous computing environments to share the same interpretation of the information exchanged.

Thirdly, individual privacy preferences towards dynamic context-aware environment are likely to consist of a complex set of rules in response to various situations and changes over time. A formal context model that is rich and flexible enough to represent a variety of context information, and be easily extendable to allow the adding of new kinds of context information that have not been anticipated during design phase, is desirable. In addition, the desirable model is expected to be capable of providing efficient reasoning and inference mechanisms, to detect inconsistency in the acquired information since context information can be highly imperfect, and to make the representation of complex and high-level abstraction of social context possible. This extensibility and reasoning capabilities can be achieved by computing-evaluable formality inherent in an ontology-based context information model.

To complement the discussion on why the ontology-based approaches should be used in this privacy protection work, it is also useful to have some form of understanding on how they may be used. The ontology-based technologies will be exploited in this work to fulfill the following tasks:

- Detecting and resolving privacy preference conflicts and redundancy.
- Performing evaluation of data collecting policies against individual semantically-rich privacy preferences.
- Conducting context inference to perceive the people context before making proper information disclosure decisions.

The author will present the ontology-based modeling of the Privacy Policy/Preference Language and context information in section 6.2 and 6.3 respectively, and describe in detail the use of ontology-based semantic reasoning to achieve the above objectives in section 6.4, 6.5 and 6.6.

6.1.2 Choice of Ontology Language

Various ontological modelling approaches and languages are analyzed and compared in [1]. The intention here has been to favour the one that is closest to a standardized solution in order to ensure architectural consistency and model interoperability for future development. Web Ontology Language (OWL) [51] developed by W3C is currently a forefront approach to the standardization of ontology languages and is expected to be widely deployed. Various tools (such as the Protégé [133] developed by Stanford University and Altova SemanticWorks [134]) exist to help develop OWL ontologies with relative ease.

OWL is based on the Resource Description Framework (RDF) Schema[13] [132], and can be used to explicitly represent the meaning of terms in vocabularies and the

[13] A technique is used for representing knowledge.

relationships between those terms. W3C states that OWL is intended to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans. In other words, the terms and relationships specified in OWL are intended to make it easier for machines to automatically process and integrate information, which matches the author's requirements for the context modelling in terms of better supporting reasoning capability.

OWL provides three expressive sublanguages [51] with different applicability.

- OWL-Lite is syntactically the simplest sub-language. It is intended to be used in situations where only a simple class hierarchy and simple constraints are needed.
- OWL-DL is more expressive than OWL-Lite and supports applications that need maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in a finite time).
- OWL-Full is the most expressive OWL sub-language; it is intended to be used in situations where maximum expressiveness and the syntactic freedom of RDF are more important than being able to guarantee the decidability or computational completeness of the language. It is therefore not possible to perform automated reasoning on OWL-Full ontologies.

To consider applying the OWL sub-languages in this work, OWL-Full is thought not to be applicable, as there is no computational guarantee; this is definitely required in the context aware ubiquitous computing environment which always has to face resource-constrained devices. The author has chosen to develop ontologies using OWL-DL. Investigation showed that OWL-Lite was insufficient to describe some of the information relations that are required to conduct semantic reasoning on privacy preferences and policies. The author employs the Protégé ontology development tool

[133] developed by Stanford University as a major ontology editor, and uses Altova SemanticWorks [134], a commercial OWL-DL editor and illustration tool, to help present some of ontology representation in this work.

6.1.3 Ontology Design Methodology and Process

The ontology design approach was roughly based on a method for engineering ontologies suggested by Noy and McGuinness in [50], consisting of the following general steps:

- Determine the domain and scope of the ontology
- Consider reusing existing ontologies
- Enumerate important terms in the ontology
- Define the classes and class hierarchy
- Define the properties of classes
- Define the restrictions of properties.
- Create instances

The development process was based on these general steps, but did not strictly follow them. In particular, when developing the Privacy Preference Rule Ontology, the second step was omitted, since it was found out that no relevant ontology exists. Also, the ontology development process was iterative in nature as expected [50].

To provide a better feel for this process, a few specific examples of developing the Privacy Preference Rule Ontology are given below (a subset of the ontological specification of the Privacy Preference Rule Ontology is illustrated in Figure 6.1):

• *Domain and scope of the ontology:* includes concepts associated with expressing

specific preference rules and privacy meta-policy, as well as concepts used to resolve preference rule conflict (e.g. Precedence)

- Enumerate important terms in the ontology: Rule, Rule Behaviour, Data, Policy Condition, Context Condition, etc.
- Define classes and class hierarchy: the class Recipient, Retention, and Purpose belong to PolicyCondition, while the class Target, Time, Location and Activity are ContextCondition (see Figure 6.1).
- *Define the class properties:* the root class *Rule* has the following properties: *hasData*, *hasBehavior*, *hasPolicyCon*, *hasContextCon* and *hasPrecedenceOver*. These properties link the root class *Rule* to other important classes, e.g. *Data*, *Behavior*, *PolicyCondition*, *ContextCondition*.
- *Define the property restrictions:* the *hasBehavior* property can only take on "*Permit*", "*Forbid*", and "*Askme*" as its attribute value, while the *hasData* property could take on any attribute value that have been defined in the base data schema of the Privacy Policy/Preference Language.
- *Create instances:* a preference rule *"my location information should not be disclosed to my colleagues when I am not working"* is created as follows: the rule *hasBehavior*="Forbid", *hasData*= "Location", *hasContextCondition*="contextcondition", contextcondition *hasTarget*="Colleagues", contextcondition *hasTime*="Offworking".

A more detailed description on the ontologies developed by the author is given in the following section 6.2 and 6.3.

6.2 Privacy Preference Rule Ontology

Chapter 5 presented human-readable data collecting policies and privacy preferences that could be encoded in XML schema. To facilitate automated processes of privacy control, the author has experimented with employing ontological modeling techniques to model privacy vocabulary, including both privacy data elements and disclosure conditions. This vocabulary can then be used by inference engines planted in the Privacy Agents to reason over ontology descriptions as a means of supporting privacy rule evaluation, as well as conflict detection and resolution.

Figure 6.1 depicts a subset of the ontological specification of the privacy preference rule syntax. Every rule is expressed with four elements: Data (*Data class*), Policy Condition *(PolicyCondition class)*, Context Condition *(ContextCondition class)* and Rule behavior (*Behavior class*). The Data class represents sensitive personal information specified in the base data scheme of the Privacy Policy/Preference Language in Chapter 5 (see Table 5.1), while the Behavior class consists of three individuals[14]: *Permit*, *Forbid*, and *Askme*, which are also specified in the language. The author uses description logic to apply constraints that every rule must specify one and only one behavior (cardinality of *hasBehavior* property =1), that it consists of one policy condition (cardinality of *hasPolicyCon* property =1) and one context condition (cardinality of *hasContextCon* property =1), and that only one data element (cardinality of *hasData* property =1) is allowed in every rule.

[14] Individuals represent objects in a domain. In the Protégé ontology tool [133], individuals are also known as instances, and can be referred to as being 'instances of classes' [166].

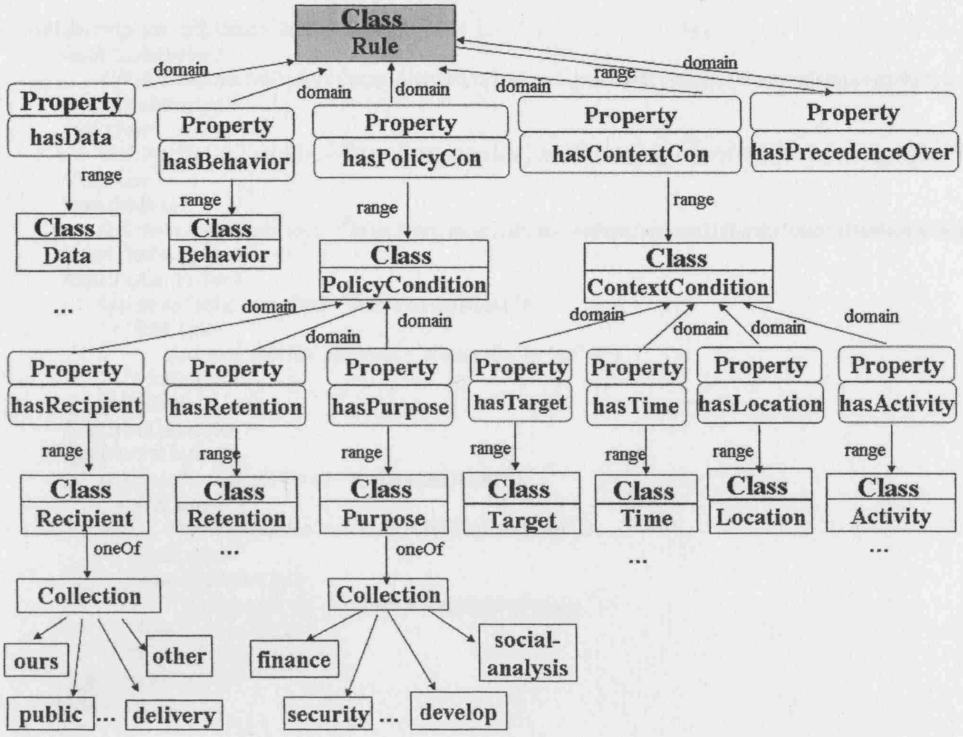

Figure 6.1 *A subset of the ontological specification of privacy preference rule ontology*

Corresponding to user-specified privacy preferences described in Chapter 5, the policy condition constrains data collecting practices to respond to their Purpose, Recipient and Retention, while the context condition is constrained to limit information disclosure to respond to specific contexts (Target, Location, Time and Activity). Every valid rule has at most one of each type of condition element (maxCardinality of hasTarget, hasLocation, hasTime, hasActivity properties =1), and each condition element could have zero or one value. Rules with empty conditions are treated as an unconditional application of a rule behavior to the data specified in the rules. Rules with empty data imply that a rule behavior is applicable to all data under the conditions specified. Figure 6.2 exemplifies a preference rule instance that is defined using description logic constraints specified above. XML representation of a full specification of the Privacy Preference Rule Ontology is accessible in http://www.ee.ucl.ac.uk/~jezhang/PrivacyPreferenceRuleOntology.owl

```
<rdf:Description rdf:about="#rule1">
    <pppl:hasBehavior>
        <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/privacypreferenceruleontology#Permit "/>
    </pppl:hasBehavior>
    <rdf:type>
        <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/privacypreferenceruleontology#Rule "/>
    </rdf:type>
    <pppl:hasData>
        <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#homecontact "/>
    </pppl:hasData>
    <pppl:hasContextCon>
        <rdf:Description rdf:about="#contextcondition1">
            <rdf:type>
                <rdf:Description rdf:about="#ContextCondition"/>
            </rdf:type>
        </rdf:Description>
    </pppl:hasContextCon>
    <pppl:hasPolicyCon>
        <rdf:Description rdf:about="#policycondition1">
            <rdf:type>
                <rdf:Description rdf:about="#PolicyCondition"/>
            </rdf:type>
            <pppl:hasRetention>
                <rdf:Description rdf:about="#legal-requirement"/>
            </pppl:hasRetention>
        </rdf:Description>
    </pppl:hasPolicyCon>
</rdf:Description>
```

Figure 6.2 *OWL specification of a preference rule instance represented in the RDF/XML scheme*

Note that any individual preference rule instances specified, based on the Privacy Preference Rule Ontology above, represent an atomic rule that is stored in the Privacy Preferences Repository and is used for policy evaluation (of data collecting policies). Being an atomic rule implies that the data specified in *Data class* is allowed to perform the behavior specified in *Behavior class* only when all the conditions specified in *Condition class* are satisfied. As a result, rules with disclosure conditions connected with an "or" connective are transformed into multiple atomic rules before they are stored in the Privacy Preferences Repository. The author has developed four principles[15] to transform user-specified privacy preferences to atomic rules that comply with ontology description of Privacy Preference Rule Ontology. Figure 6.3 below gives an example showing that two user-specified rules in the preference 2 in Chapter 5 (see Figure 5.5) are transformed into three atomic rules represented in RDF/XML scheme.

[15] The four transformation principles will be presented in the proof-of-concept implementation in Chapter 7.

This transformation approach helps maintain the flexibility and a good expressiveness of user-specified privacy preferences, while at the same time addressing a key problem. The problem is that the ontology-based description logic is inept at dealing with a logic operand like “or” and “and”.

```
<rdf:Description rdf:about="#rule1-pref2">
    <rdf:type>
        <rdf:Description rdf:about="#Rule"/>
    </rdf:type>
    <pppl:hasBehavior>
        <rdf:Description rdf:about="#Permit"/>
    </pppl:hasBehavior>
    <pppl:hasContextCon>
        <rdf:Description rdf:about="#contextcondition1_pref2">
            <rdf:type>
                <rdf:Description rdf:about="#ContextCondition"/>
            </rdf:type>
            <pppl:hasTarget>
                <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#colleague"/>
            </pppl:hasTarget>
            <pppl:hasActivity>
                <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/activityontology#working">
                    <rdf:type>
                        <rdf:Description rdf:about="http://www.w3.org/2002/07/owl#Class"/>
                    </rdf:type>
                </rdf:Description>
            </pppl:hasActivity>
        </rdf:Description>
    </pppl:hasContextCon>
    <pppl:hasData>
        <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#location">
            <rdf:type>
                <rdf:Description rdf:about="http://www.w3.org/2002/07/owl#Class"/>
            </rdf:type>
        </rdf:Description>
    </pppl:hasData>
    <pppl:hasPolicyCon>
        <rdf:Description rdf:about="#policycondition1_pref2"/>
    </pppl:hasPolicyCon>
</rdf:Description>
<rdf:Description rdf:about="#rule2_pref2">
    <pppl:hasBehavior>
        <rdf:Description rdf:about="#Permit"/>
    </pppl:hasBehavior>
    <pppl:hasData>
        <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#location"/>
    </pppl:hasData>
    <pppl:hasContextCon>
        <rdf:Description rdf:about="#contextcondition2_pref2">
            <rdf:type>
                <rdf:Description rdf:about="#ContextCondition"/>
            </rdf:type>
            <pppl:hasActivity>
                <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/activityontology#notworking"/>
            </pppl:hasActivity>
```

```
            <pppl:hasTarget>
                <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#family"/>
            </pppl:hasTarget>
        </rdf:Description>
    </pppl:hasContextCon>
    <rdf:type>
        <rdf:Description rdf:about="#Rule"/>
    </rdf:type>
    <pppl:hasPolicyCon>
        <rdf:Description rdf:about="#policycondition2_pref2"/>
    </pppl:hasPolicyCon>
</rdf:Description>
<rdf:Description rdf:about="#rule3_pref2">
    <rdf:type>
        <rdf:Description rdf:about="#Rule"/>
    </rdf:type>
    <pppl:hasBehavior>
        <rdf:Description rdf:about="#Permit"/>
    </pppl:hasBehavior>
    <pppl:hasContextCon>
        <rdf:Description rdf:about="#contextcondition3_pref2">
            <pppl:hasActivity>
                <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/activityontology#notworking"/>
            </pppl:hasActivity>
            <rdf:type>
                <rdf:Description rdf:about="#ContextCondition"/>
            </rdf:type>
            <pppl:hasTarget>
                <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#friend"/>
            </pppl:hasTarget>
        </rdf:Description>
    </pppl:hasContextCon>
    <pppl:hasData>
        <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#location"/>
    </pppl:hasData>
    <pppl:hasPolicyCon>
        <rdf:Description rdf:about="#policycondition3_pref2"/>
    </pppl:hasPolicyCon>
</rdf:Description>
```

Figure 6.3 *RDF/XML representation of atomic rules transformed from the preference 2*

The Privacy Preference Rule Ontology also supports various information relations by defining eleven properties: equalTo_d, subsume_d, subsumedBy_d, equalTo_p contain_p, containedBy_p, equalTo_c contain_c, containedBy_c, mutuallySubsume_c and mutuallySubsumedBy_c. These properties are used to conduct policy analysis on disclosure data, policy conditions, and on context conditions of preferences rules. Table 6.1 summarizes the Description Logic restrictions that are defined in the Privacy Preference Rule Ontology for each property. According to the OWL specification [51], the *domain* of a property limits individuals to which the property can be applied. If a property relates an individual to another individual, and the property has a class as one

of its domains, then the individual must belong to that class. The *range* of a property limits the individuals that the property may have as its value. If a property relates an individual to another individual, and the property has a class as its range, then the other individual must belong to the range class. Being a *Symmetric* property means if a pair (x,y) is an instance of the symmetric property P, then a pair (y,x) is also an instance of P. Being a *Transitive* property means if the pair (x,y) is an instance of the transitive property P, and the pair (y,z) is an instance of P, then the pair (x,z) is also an instance of P. One property may be stated to be the *Inverse Of* another property. If the property P1 is stated to be the inverse of the property P2, then if X is related to Y by the P2 property, then Y is related to X by the P1 property.

Table 6.1 *Description Logic restrictions that are defined in the Privacy Preference Rule Ontology on each property*

Properties	**Property Characteristics**	**Inverse of**	**Domain**	**Range**
equalTo_d	*SymmetricProperty* *TransitiveProperty*	-	Data	Data
subsume_d	*TransitiveProperty*	subsumedBy_d	Data	Data
subsumedBy_d	*TransitiveProperty*	subsume_d	Data	Data
equalTo_p	*SymmetricProperty* *TransitiveProperty*	-	Policy Condition	Policy Condition
Contain_p	*TransitiveProperty*	containedBy_p	Policy Condition	Policy Condition
containedBy_p	*TransitiveProperty*	Contain_p	Policy Condition	Policy Condition
equalTo_c	*SymmetricProperty* *TransitiveProperty*	-	Context Condition	Context Condition
Contain_c	*TransitiveProperty*	containedBy_c	Context Condition	Context Condition
containedBy_c	*TransitiveProperty*	Contain_c	Context Condition	Context Condition
mutuallySubsume_c	*SymmetricPropert* *TransitiveProperty*	mutuallySubsumedBy_c	Context Condition	Context Condition
mutuallySubsumedBy_c	*SymmetricProperty* *TransitiveProperty*	mutuallySubsume_c	Context Condition	Context Condition

The ***equalTo_d*** property is used to describe that the data elements of two preference

rules have the same value, while the ***subsume_d*** and ***subsumedBy_d*** relations describe subsumption relation of two data elements. For example, the expression "DataA subsume_d DataB" means that DataB is a subclass of DataA in the ontology hierarchy. If the data element of a preference rule does not have any value, the preference rule is applicable to all data under the conditions it specifies, and thus subsumes other preference rules that have specific values on their data element.

The ***equalTo_p*** property specifies that the policy conditions of two preference rules are equal (i.e. two preference rules have equivalent policy condition). Since policy condition constrains data collecting practices to respond to their Purpose, Recipient and Retention, having equivalent policy condition means that two preference rules have identical policy condition elements (i.e. the same combination of Purpose, Recipient and Retention elements), and values of correspondent elements in two rules (if existing) are equal. The attribute values of Purpose, Recipient and Retention classes are defined following the P3P practice and are specified in the Privacy Policy/Preference Language presented in Chapter 5.

The ***contain_p*** and ***containedBy_p*** properties describe that the policy condition of one preference rule contains that of another. By containing, it means that the application scope of one policy condition covers that of another. In other words, the former has less stringent constrains on its policy condition. A more concrete example is that if preference rule A has a *Purpose* element with the value "develop" and the *Recipient* element with the value "ours", while preference rule B only has *Purpose* element with value "develop", then it says that the policy condition of preference rule B contains that of preference rule A; preference rule B has less stringent constrains on its policy condition, and thus covers a wider application scope than preference rule A. The ***containedBy_p*** property is owl:inverseOf of ***contain_p*** property.

The ***equalTo_c*** property specifies that the context conditions of two preference rules are equal (i.e. two preference rules have the equivalent context condition). Since the context

condition is constrained to limit information disclosure to respond to specific contexts (Target, Location, Time and Activity), having equivalent context condition means that two rules have identical context condition elements (i.e. the same combination of Target, Location, Time and Activity elements), and values of correspondent elements in two rules (if existing) are equal.

The ***contain_c*** and ***containedBy_c*** properties describe that context condition of one preference rule contains that of another. Similar to the definition of ***contain_p*** and ***containedBy_p*** properties, the containing means that the application scope of one context condition covers that of another. In other words, the former has less stringent constrains on its context condition. Different from the definition of the ***contain_p*** and ***containedBy_p*** properties, there are basically two situations to specify that one context condition contains another.

- Firstly, context condition of two preferences rules have identical context condition elements (i.e. the same combination of Target, Location, Time and Activity elements), and one context condition has at least one element whose value is subsumed by (or subsumes) that of the corresponding element in another context condition while all other elements may have subsumed or equal value to their corresponding elements in another context condition. For instance, a preference rule A has *Target* element with value "colleagues" and *Time* element with value "workinghours", while a preference rule B has *Target* element with value "socialgroup", and *Time* element with value "workinghours", then it says that context condition of preference rule B contains that of preference rule A, as according to the ontology description of the Privacy Preference Rule Ontology, "colleagues" is a subset of "socialgroup".
- Secondly, context conditions of two preferences rules do not have identical context condition elements, and one context condition includes all context elements defined in another context condition, with their values either subsume or equal to that of the corresponding element in another context condition. For instance, if preference rule A has *Target* element with value "colleagues" and *Time* element with value

"workinghours", while preference rule B only has *Target* element either with value "colleagues" or "socialgroup", then it says that context condition of preference rule B contains that of preference rule A. In both situations, the preference rule B has less stringent constrains on its context condition, and thus covers a wider application scope than preference rule A. The ***containedBy_c*** property is owl:inverseOf of ***contain_c*** property.

The ***mutuallySubsume_c*** and ***mutuallySubsumedBy_c*** properties are used to describe more complex cases in that the context condition of two preference rules have (at least two) identical condition elements (Target, Location, Time, and Activity), and one context condition has at least one element whose value subsume that of the corresponding element in another context condition and has at least another element whose value is subsumed by that of the corresponding element in another context condition. A concrete example is that if preference rule A has *Target* element with value "colleagues" and *Time* element with value "anytime", while preference rule B has *Target* element with value "socialgroup", and *Time* element with value "workinghours", then it says that context condition of preference rule A mutually subsume that of preference rule A. In this case, it is impossible to tell which preference rule has more stringent constrains on its context condition, and that the application scope of one preference rule covers that of another.

The eleven properties will be used to reason over the ontology description of the Privacy Preference Rule Ontology as a means to conduct semantic policy analysis and reasoning. The author will present in section 6.4 and 6.5 how inference engines planted in the Privacy Agent can employ various concepts and relations defined in the Privacy Preference Rule Ontology to detect privacy preference conflicts and redundancy, and to conduct policy evaluation respectively.

6.3 Context Information Modeling

In the author's proposal, Privacy Agent needs to perceive its owner's context, in order to make appropriate information disclosure decision for him/her. In Chapter 2, the author has followed Dey's approach and has scoped the context information relevant to personal privacy concerns in this work as any information (obtained either explicitly or implicitly) that can be used to characterize privacy aspects of a person. The author's work on context information modeling enumerates four primary contexts related to human users, including personal information (e.g. identity, personal profile such as gender, etc.), location, time and activity. The four primary contexts act as indices to form other sorts of contextual information, and are staple elements to express individual privacy preferences. For instance, a person may specify that she wants to let her colleagues (Personal Information) see where she is (Location) or access her calendar activities (Activity) between 8am and 5pm on weekdays but not over the weekend (Time). Based on this selection of context information, four sub-ontologies are developed, i.e. Personal Information Ontology, Location Ontology, Time Ontology, and Activity Ontology. An alternative is to have a single "huge" Context Ontology having all the terms and their relationships that are specified separately in the four sub-ontologies. This might work in a demonstration scenario, but would lead to severe management and maintenance challenges in real world situations.

Figure 6.4 below illustrates relationships among the four contextual sub-ontologies and their relationships with the Privacy Preference Rule Ontology. The Data class specified in the Privacy Preference Rule Ontology represents sensitive personal information that is enclosed in the Personal Information Ontology, while Target, Location, Time, and Activity classes in the Privacy Preference Rule Ontology describe the context conditions of the information disclosure practice, and are represented with concepts defined in the Personal Information Ontology, Location Ontology, Time Ontology, and Activity Ontology respectively. The Personal Information Ontology includes not only static but also dynamic context information like a person's current location (Location

Ontology) and activity (Activity Ontology), and the inference process to obtain dynamic context information requires the use of concepts and relations defined in the Location Ontology, Time Ontology and Activity Ontology. The Activity Ontology captures human activities, such as working, traveling, studying, which often happen in a certain time span (Time Ontology) and location (Location Ontology). The work uses owl:import syntax to present the relationships among different ontologies.

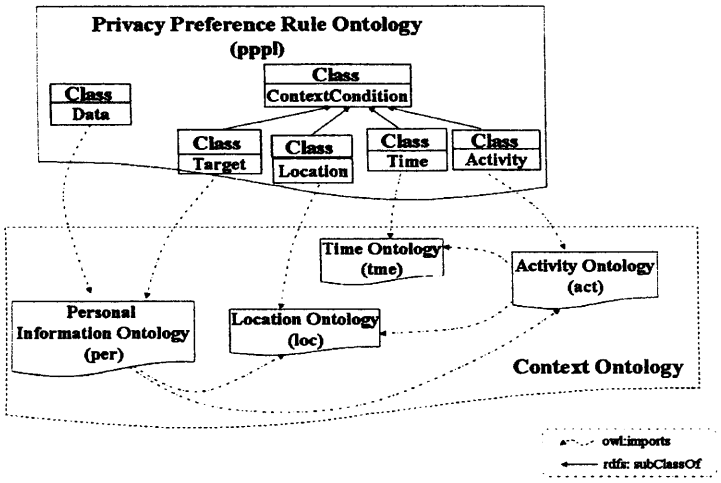

Figure 6.4 *An overview of relationships between the Context Ontology and Privacy Preference Rule Ontology, and relationships among the four contextual sub-Ontologies*

A key strength of working with ontologies is that developers can reuse work done by other ontologists by importing their published ontologies. The development of context ontology has benefited from DAML-Time [135] and OWL-Time [136], and the spatial ontologies in OpenGIS [137]. However, the development strategy adopted by this work is to borrow relevant concepts and definition of relationships from these ontologies, but not to import them directly. Although the semantics for importing ontologies is well defined, by choosing not to use this approach, it can effectively limit the overhead; the reasoning engines are not then required to import ontologies that may be irrelevant to ubiquitous computing applications. Also note that, the author only includes in each contextual sub-ontology relevant concepts and relations that are sufficient for demonstration and simulation purpose. This is in contrast to existing

context ontologies such as [48, 138, 139, 149, 170] that intend to capture as many concepts and relations as possible in an intended domain. For better interoperability and extensibility, the Context Ontology developed by the author could be extended by others or merged with other ontologies using standard OWL ontology mapping constructs (e.g. owl: equivalentClass and owl:equivalentProperty). The following sections present the development of four contextual sub-ontologies separately.

- ***Personal Information Ontology***

The Personal Information Ontology consists of sensitive data that people resort to protection; it has a direct correspondence to the base data schema of the Privacy Policy Preference Language specified in Chapter 5. The Personal Information Ontology organizes sensitive personal information in a hierarchical way so that ontology-based description logic can be used to reason over hierarchical structuring of the information as a means to facilitate semantic matching. This makes it possible to control information disclosure to various levels of accuracy (i.e. ambiguous disclosure). A concrete example is that individuals could choose to disclose their location information at city, district, or street level. Figure 6.5 below depicts a subset of such a hierarchy, and includes sensitive information that may be often used in privacy preferences and are used in the author's simulation and evaluation work.

As shown in the Personal Information Ontology, sensitive personal information may either be static, such as personal profile information (e.g. gender, age and date of birth etc.), or dynamic, such as a person's location and activities. While static information may be input manually by their owner and hardly changed once input, dynamic information is context-sensitive and various methods exist to acquire it from multiple sources. This work refers to the process of perceiving a person's dynamic contextual information as context reasoning. This process may involve reasoning over additional spatial, temporal and activity concepts and information relationships. As a result,

sensitive information included in the Personal Information Ontology is associated with relevant concepts defined in the Location Ontology, Time Ontology and Activity Ontology. The association is indicated in Figure 6.5 using double arrows and highlighted in orange. The author will present the context reasoning process in section 6.6.

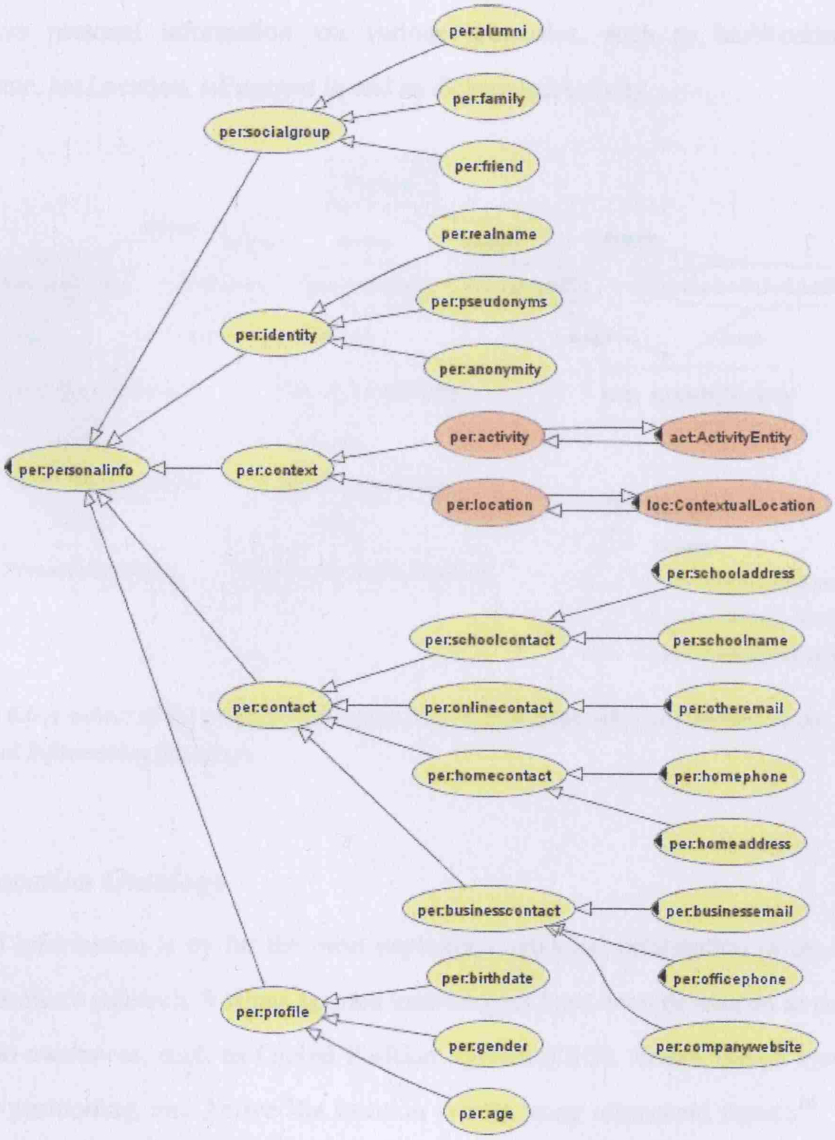

Figure 6.5 *A subset of the ontology specification of personal information (**per**: Personal Information Ontology, **act**: Activity Ontology)*

In addition to the hierarchical organization of miscellaneous personal data, the Personal

Information Ontology defines various properties to represent relations among these data. Figure 6.6 below exemplifies some properties that are defined in the Personal Information Ontology and used in the author's simulation and evaluation work on context reasoning. As illustrated in Figure 6.6, the Personal Information Ontology has a root class *Person* that represents any human users. A person links to his/her sensitive personal information via various properties, such as hasWorkingPlace, hasHome, hasLocation, isEngaged in and hasScheduledActivity.

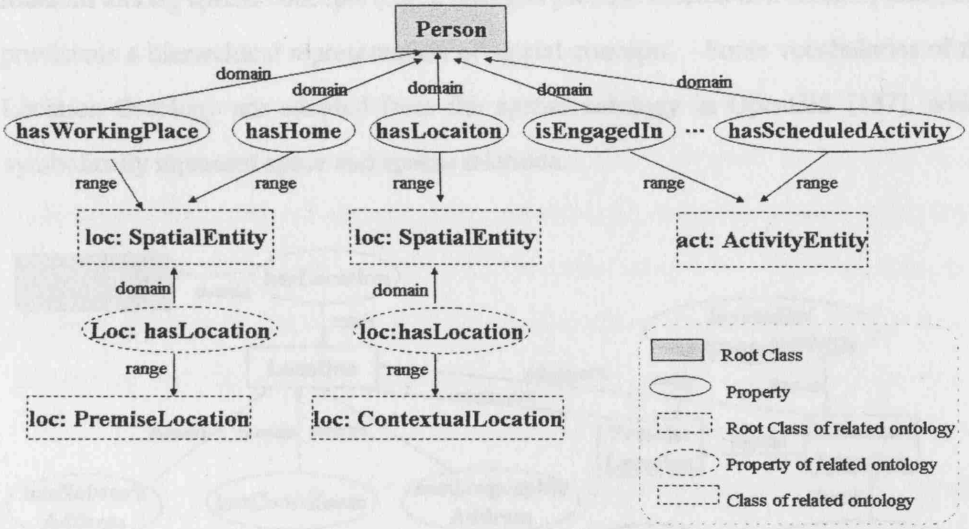

Figure 6.6 *A subset of the ontology specification of information relations defined in the Personal Information Ontology*

- ***Location Ontology***

Spatial information is by far the most exploited contextual information in the field of context-aware research. Various sensing technologies have been developed to provision location-awareness, such as Global Position System (GPS), Active Badge system for indoor positioning, and Active Bat location system using ultrasound signals[16]. These sensing technologies, with their inherent strengths and weaknesses, are likely to work together in order to provision truly ubiquitous scenarios. This leads to heterogeneous

[16] A survey and taxonomy of location systems for mobile-computing applications that describes a spectrum of current products and explores the latest research in the field is available in [131].

formats of location information and representation at various levels of accuracy.

Location Ontology developed in this work attempts to capture spatial concepts that are used by human users when specifying their privacy preferences, and by the Privacy Agent to properly perceive a person's contexts when enforcing information disclosure decisions according to individual privacy preferences. It provides mapping between spatial concepts of various formats and representations, supports reasoning about spatial relations among spatial concepts (e.g. a business premise located in a certain place), and provisions a hierarchical representation of spatial concepts. Some vocabularies of the Location Ontology are adopted from the spatial ontology in OpenGIS [137], which symbolically represent space and spatial relations.

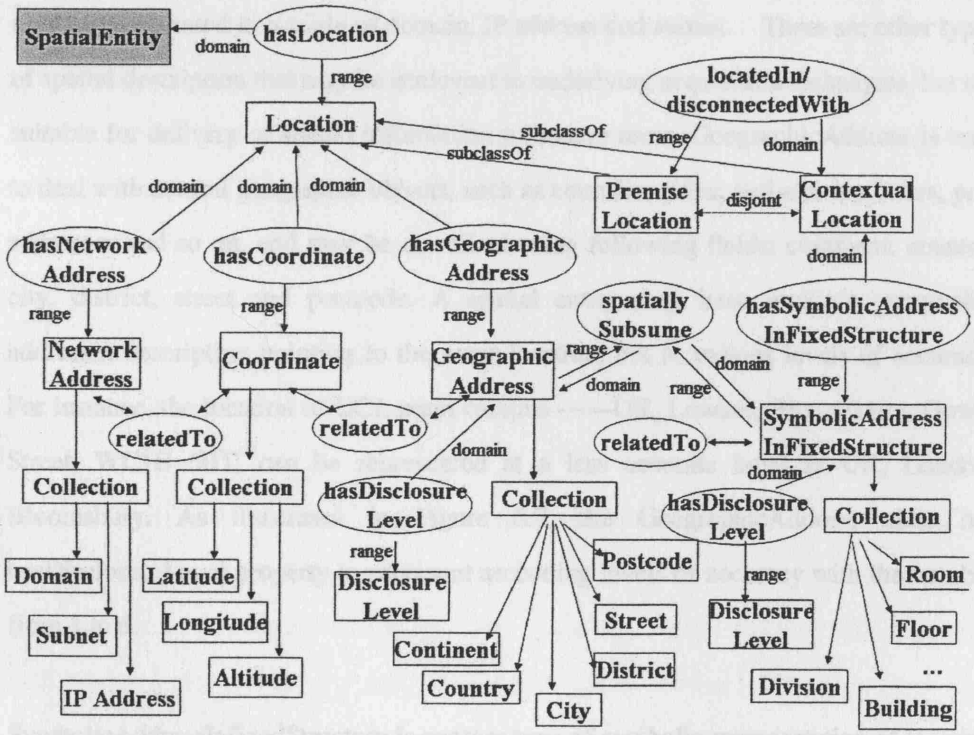

Figure 6.7 *A subset of the ontology specification of location context*

As an example, Figure 6.7 above illustrates the ontology specification of the Location context. Only a subset of the terms and their relationships specified in the Location Ontology are presented. As illustrated in the figure, Spatial Entity is a root element of

the Location Ontology. It represents spatial objects that have location information, such as a person, a company, a theater, a university, a person's home, etc. The Spatial Entity has *hasLocation* property to link it to its location. The *Location* class has two subclasses: *PremiseLocation* and *ContextualLocation*. The former consists of relatively static premise location, such as a company's location and a person's home, while the latter includes dynamic location of moving entities, such as a person's current location.

The Location Ontology supports various formats of spatial information, some of them are in direct correspondence to sensing technologies that acquire them. For instance, *Coordinate* subclass are represented as a <latitude, longitude> pair (an altitude coordinate can be added if necessary) and are typically obtained via Global Position Systems (GPSs), while the network address obtained through IP network technologies may be represented as a triple of domain, IP address and subnet. There are other types of spatial description that may be irrelevant to underlying acquisition techniques, but are suitable for delivery of spatial information to human users. GeographicAddress is used to deal with natural geographic objects, such as counties, cities, and also zip codes, post addresses and so on, and may be described using following fields: continent, country, city, district, street and postcode. A spatial entity may have multiple geographic addresses description pointing to the same location, but at various levels of accuracy. For instance, the location of UCL main campus ——UK, London, Bloomsbury, Gower Street, WC1E 6BT, can be represented at a less accurate level as UK, London, Bloomsbury. As illustrated in Figure 6.7, the GeographicAddress class has hasDisclosureLevel property to represent ascending levels of accuracy with the number from 1 to 6.

SymbolicAddressInfixedStructure is another type of symbolic representation of location data. The Location Ontology provides flexible constructs to support the symbolic description of location in application-specific spatial structures, such as campus, department store, library, etc. Context-aware applications provisioned in SmartSpace environments are typically required to deal with this format of spatial information. Like

the geographic address, the symbolic location in fixed structures can be represented at various levels of accuracy. The accuracy depends on the hierarchy of specific structures. For instance, in a SmartCampus environment like e-Campus in [17], a person's location may be represented using division, building and office room, while in a department store, spatial concepts such as floor and section are used to describe a user's location.

The Location Ontology supports various spatial relations by defining five properties: *relatedTo*, *locatedIn*, *disconnectedWith*, *spatiallySubsume* and *spatiallySubsumedBy* properties. The *relatedTo* property is used to associate multiple location data that represent the same place. The *locatedIn* and *disconnectedWith* property are used to describe that a contextual location (e.g. a person's current location) is within or outside a premise location (e.g. campus). The *spatiallySubsume* and *spatiallySubsumedBy* properties describe hierarchy of spatial concepts in GeographicAddress and SymbolicAddressInfixedStructure classes. For example, country spatially subsumes city that in turn subsumes a building (in GeographicalAddress class), and building spatially subsumes room (in SymbolicAddressInfixedStructure class).

Table 6.2 summarizes the Description Logic restrictions that are defined in the Location Ontology for each property, while Figure 6.8 exemplifies the ontology specification of *locatedIn* property using a RDF/XML representation.

The author will discuss in section 6.6 how various concepts and relations defined in the Location Ontology are employed to reason over the spatial information of a person location context (i.e. spatial reasoning). The spatial reasoning is a staple process when the Privacy Agent perceives a person's context before making information disclosure decisions.

Table 6.2 *Description Logic restrictions that are defined in the Location Ontology on each property*

Properties	Property Characteristics	Inverse of	Domain	Range
RelatedTo	*SymmetricProperty* *TransativeProperty*	--	Network Address	Coordinate
			Coordinate	Network Address
			Coordinate	Geographic Address
			Geographic Address	Coordinate
			Geographic Address	Network Address
			Network Address	Geographic Address
LocatedIn	*TransitiveProperty*	disconnectedWith	Contextual Location	Premise Location
disconnectedWith	*TransitiveProperty*	LocatedIn	Contextual Location	Premise Location
spatiallySubsume	*TransitiveProperty*	spatiallySubsumedBy	Geographic Address	Geographic Address
			Symbolic AddressInFixedStructure	Symbolic AddressInFixedStructure
spatiallySubsumedBy	*TransitiveProperty*	SpatiallySubsume	Geographic Address	Geographic Address
			Symbolic AddressInFixedStructure	Symbolic AddressInFixedStructure

```
<owl:TransitiveProperty rdf:ID="locatedIn">
    <owl:inverseOf>
        <owl:TransitiveProperty rdf:ID="disconnectedWith"/>
    </owl:inverseOf>
    <rdfs:domain rdf:resource="#PremiseLocation"/>
    <rdfs:range rdf:resource="#ContextualLocation"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</owl:TransitiveProperty>
```

Figure 6.8 *Ontology specification of locatedIn property using a RDF/XML representation*

- ***Time Ontology***

Temporal information is another important category of contextual information. Many human activities in ubiquitous computing environments are time-sensitive, and happen only in a certain period or at a point of time. Also, individual privacy requirements may change over time and in response to context changes. Time Ontology is developed to capture abstract temporal concepts, such as working hours, during holiday, off-working time, weekend, that may be used by individuals to express their privacy preferences, and to establish mappings between these abstract temporal concepts and exact time description using seconds, minutes, hours, days, and so on. The development of the Time Ontology here has gained from previous work, in particular, DAML-Time ontology by Hobbs et al. [135] and OWL time ontology developed by Pan [136]. The DAML-Time Ontology is a rich collection of temporal concepts and axioms, and intended to be a complete specification of a theory of time as required for Semantic Web applications. Pan's time ontology is a simplified version of DAML-Time using OWL expression. Like Pan's OWL time ontology, the Time Ontology developed by this work only comprises some basic temporal concepts and relationships that are specified in the DAML-time. and adopts a time-date description method used in OWL time ontology to represent abstract temporal concepts using exact time descriptions.

As an example, Figure 6.9 below illustrates the ontology specification of time context. Only a subset of concepts and their relations specified in the Time Ontology are presented. As illustrated in the figure, the Time Ontology has a root class TemporalEntity which represents both exact time and abstract temporal concepts. These abstract temporal concepts, such as working day, weekend, holiday, are often used by individuals to express their privacy preferences, and can be further described using basic temporal unit, such as seconds, minutes, day of week, dates, month, year, etc.

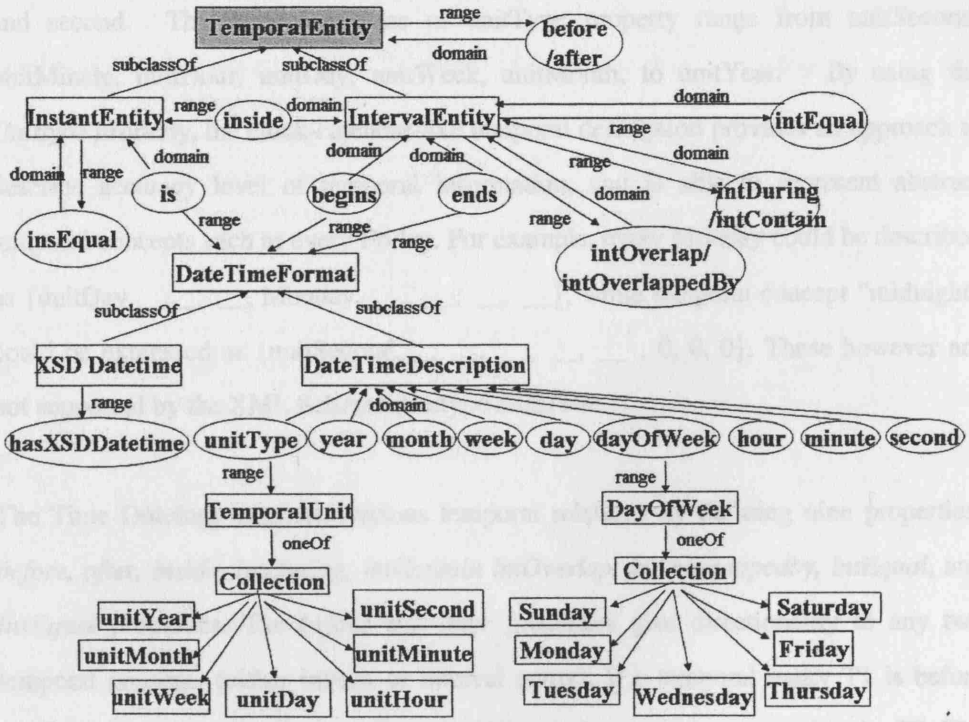

Figure 6.9 *A subset of the ontology specification of time context*

Temporal concepts are categorized to be either instant (InstantEntity subclass) or interval (IntervalEntity subclass). The instant entity represents, intuitively, point-like concepts, such as midnight, whereas the interval entity category includes temporal concepts that last a certain period of time, such as holiday (lasting few days or weeks), working-hour (from 9:00 to 17:00), working day (from Monday to Friday). In the Time Ontology specification, the instant entity uses *is* property to link itself to actual time description (i.e. temporal description classes), while interval entity is defined using *begins* and *ends* properties.

The Time Ontology provides two ways to express actual time description: either using standard XML Schema datatype *dateTime*[17] or using a clock-calendar-like description proposed by the OWL-Time ontology [136]. The calendar-clock temporal description has the following fields: unitType, year, month, week, day, dayOfweek, hour, minute

[17] http://www.w3.org/TR/xmlschema11-2/#dateTime

and second. The attribute values of unitType property range from unitSecond, unitMinute, unitHour, unitDay, unitWeek, unitMonth, to unitYear. *By* using the *Unittype property*, the clock-calendar-like temporal description provides an approach to describe accuracy level of temporal information, and is able to represent abstract temporal concepts such as every Friday. For example, every Monday could be described as {unitDay,__,__, __, Monday, __, __, __, __}, while temporal concept "midnight" could be expressed as {unitSecond,__, __, __, __, __ , 0, 0, 0}. These however are not supported by the XML Schema datatype *dateTime*.

The Time Ontology supports various temporal relations by defining nine properties: *before, after, inside, intDuring, intContain intOverlap, intOverlappedby, intEqual,* and *insEqual* properties. The *before* and *after* properties give directionality to any two temporal concepts (either instant or interval entity). If a temporal entity T1 is before another temporal entity T2, then the end of T1 is before another temporal entity T2. The *inside* property describes temporal relation between instant entity and interval entity, while *intDuring, intContain, intOverlap,* and *intOverlapedBy* are used to represent relations between two interval entities. The *intEqual* and *insEqual* properties are used between interval entities and instant entities respectively. They are able to associate multiple temporal concepts that represent the same time. Table 6.3 summarizes the Description Logic restrictions that are defined in the Time Ontology for each property, while Figure 6.10 exemplifies the ontology specification of *intContain* and *intEqual* properties using a RDF/XML representation.

The author will discuss in section 6.6 how various concepts and relations defined in the Time Ontology are employed to conduct temporal reasoning. The temporal reasoning is a staple process when the Privacy Agent perceives a person's context before making information disclosure decisions.

Table 6.3 *Description Logic restrictions that are defined in the Time Ontology on each property*

Properties	Property Characteristics	Inverse of	Domain	Range
Before	TransitiveProperty	After	Temporal Entity	Temporal Entity
after	TransitiveProperty	Before	Temporal Entity	Temporal Entity
Inside	TransitiveProperty	--	Instant Entity	Interval Entity
intDuring	TransitiveProperty	intContain	Interval Entity	Interval Entity
intContain	TransitiveProperty	IntDuring	Interval Entity	Interval Entity
intOverlap	SymmetricProperty	intOverlappedBy	Interval Entity	Interval Entity
intOverlappedBy	SymmetricProperty	intOverlap	Interval Entity	Interval Entity
intEqual	SymmetricProperty TransitiveProperty	--	Interval Entity	Interval Entity
insEqual	SymmetricProperty TransitiveProperty	--	Instant Entity	Instant Entity

```
<owl:TransitiveProperty rdf:ID="intContain">
    <rdfs:range rdf:resource="#IntervalEntity"/>
    <owl:inverseOf rdf:resource="#intDuring"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="#IntervalEntity"/>
</owl:TransitiveProperty>

<owl:TransitiveProperty rdf:about="#intEqual">
    <rdfs:domain rdf:resource="#IntervalEntity"/>
    <rdfs:range rdf:resource="#IntervalEntity"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#SymmetricProperty"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</owl:TransitiveProperty>
```

Figure 6.10 *Ontology specification of intContain and intEqual properties using a RDF/XML representation*

- ***Activity Ontology***

Activity Ontology is developed to capture human activities like working, traveling, studying, and represents them in a hierarchical way. Figure 6.11 illustrates the ontology specification of activity context. Only a subset of concepts and their relations specified in the Activity Ontology are presented. As illustrated in the figure, the Activity Ontology has a root class ActivityEntity which represent any kinds of human

activities. The activity entity can be categorized by the time they happen (PastActivity or OngoingActivity) and whether they are scheduled or not (ScheduledActivity or unscheduledActivity). Both past activities and ongoing activities could be either scheduled or unscheduled. Some of scheduled activities might never happen. The history of activity may be recorded to make available personalized services or applications that are based on history data.

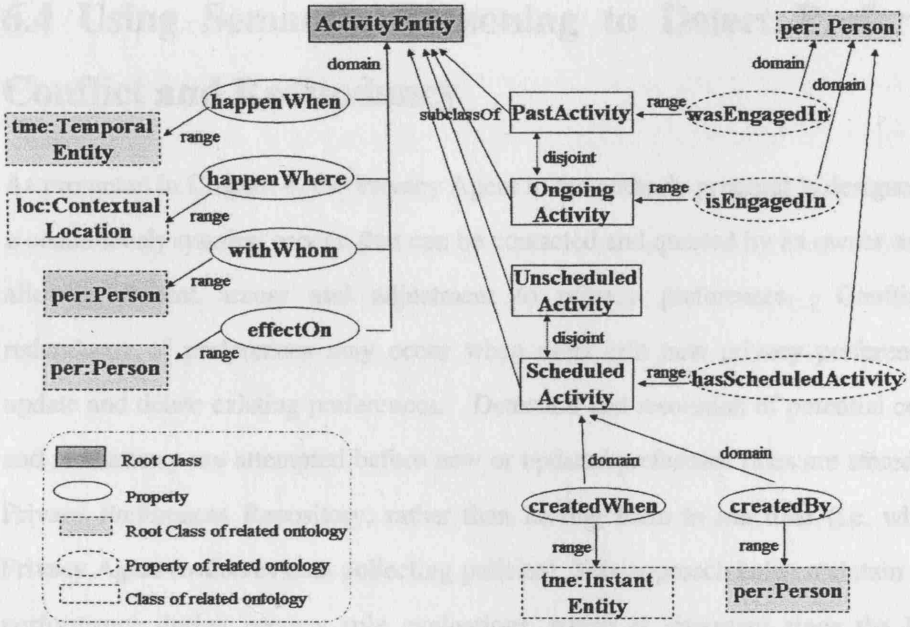

Figure 6.11 *A subset of the ontology specification of activity context*

Unlike other context categories such as time and location context, the information about human activities is not fact-based, but situation-based. In order words, it represents a higher-level abstract description of real-world situations, such as "a person is in a meeting". This cannot be obtained directly from sensing environments, and is normally derived from various individual facts of location, time, personal information, and so on. As a result, perceiving human activities may involve reasoning over temporal and spatial information, and retrieving personal information. As illustrated in Figure 6.11, the Activity Ontology is related to the Personal Information Ontology, Location Ontology, and Time Ontology by defining various properties, such as withWhom,

effectOn, happenWhere, happenWhen, etc. The author will present in section 6.6 how various concepts and relations defined in the Activity Ontology are employed by the Privacy Agent to combine reasoning over temporal and spatial information and to retrieve personal information, in order to perceive a person's activity context.

6.4 Using Semantic Reasoning to Detect Preference Conflict and Redundancy

As presented in Chapter 4, the Privacy Agent in the author's proposal is designed to be a continuously running service that can be contacted and queried by its owner anytime, allowing instant access and adjustment to privacy preferences. Conflict and redundancy of preferences may occur when users edit new privacy preferences, or update and delete existing preferences. Detection and resolution of potential conflicts and redundancy are attempted before new or updated preference rules are stored in the Privacy Preferences Repository, rather than leaving them to run time (i.e. when the Privacy Agent evaluates data collecting policies). This approach helps maintain system performance during privacy rule evaluations, which is important since the Privacy Agent is likely to work with resource-constrained devices. In this section, the author presents how to employ concepts and information relationships defined in the Privacy Preference Rule Ontology to conduct semantic policy analysis, in order to detect the conflict and redundancy of privacy preferences. The resolution methods of the preference conflict and redundancy are also discussed.

6.4.1 Detecting Preference Rule Conflict and Redundancy

Two broad classes of preference conflicts are identified in this work: *Modality Conflict* and *Non-Modality Conflict*. According to the definition of *Modality Conflict* in [140], the *Modality Conflict* in privacy preferences occurs when preference rules applied to the

same data and disclosure conditions have different rule behaviors. For instance, a user states that she would like to accept any requests for her real identity for social-analysis purpose, but later on she specifies a new rule to reject such access. *Non-Modality Conflict* results from subsets of relationships existing in data and/or conditions comprising two or more rules. For instance, a user stated that any requests for his contact information is allowed, but later on he specifies that access to home-contact information (a subset of contact information, as showed in Table 5.1 in Chapter 5) is forbidden. *Modality Conflict* differs from *Non-Modality Conflict* in that a deterministic decision must be made to follow one of the conflicting rules while discarding the other. In contrast, the conflicting rules of *Non-Modality Conflict* may co-exist under certain condition. In the example above, two preference rules could coexist, as the conflicting interest arises only when the user's homecontact information is requested by applications. For other types of contact information (e.g. schoolcontact and businesscontact), the previously specified preference remains applicable.

During the analysis of privacy reference, it is found out that the *Non-Modality Conflict* happens more often than the *Modality Conflict*, due to semantic overlap of privacy preferences. Figure 6.12 depicts a conflict profile summarizing situations that could lead to *Modality Conflict and Non-Modality Conflict.* R_{new} stands for a newly specified atomic rule, while R_{old} is an atomic rule already stored in the *Privacy Preferences Repository* in the Privacy Agent. The "$\subset$" and "$\supset$" symbols indicate subsume_d and subsumedBy_d relations of data elements, or contain_c and containedBy_c relations of context conditions as defined in the Privacy Preference Rule Ontology (see section 6.2). For instance, the "R_A.data $\subset$ R_B.data" expression means that the data value of rule A is a subset of that of rule B. The expression "R_A.contextcondition $\subset$ R_B.contextcondition" implies that context condition of rule B contains that of rule A, while the expression "R_A.contextcondition $\subset\supset$ R_B.contextcondition" stands for the mutually subsumed relation between context conditions as defined in the Privacy Preference Rule Ontology. In the aforementioned example where a person stated that any requests for his contact information is allowed, but later on specifies that the access

to home-contact information is forbidden, a non-Modality conflict (type (1) in conflict profile) occurs. The two preference rules may coexist when the newly specified rule has precedence over the old one. In this case, the new preference rule applies only to home-contact information, while the previously specified preference rule applies to all other contact information except for home-contact.

Modality Conflict (1)

(1)R_{new}.data = R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition = R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

Non-Modality (11)

(1)R_{new}.data ⊂ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition = R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(2)R_{new}.data ⊃ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition = R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(3)R_{new}.data = R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊂ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(4)R_{new}.data = R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊃ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(5)R_{new}.data ⊂ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊂ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(6)R_{new}.data ⊃ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊃ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(7)R_{new}.data ⊃ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊂ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(8)R_{new}.data ⊂ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊃ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(9)R_{new}.data = R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊂ ⊃ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(10)R_{new}.data ⊃ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊂ ⊃ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

(11)R_{new}.data ⊂ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition ⊂ ⊃ R_{old}.contextcondition
R_{new}.behvaior ≠ R_{old}.behavior

Figure 6.12 *Conflict profile of Modality and NonModality Conflict*

Figure 6.13 below summarizes situations that could lead to redundancy of preference

rules. Redundancy refers to duplicates of preference rules. Two preference rules are redundant if the application scope of one preference rule (in terms of its data and disclosure conditions) fully covers that of another. Redundancy is resolved each time new preference rules are to be updated into the *Privacy Preferences Repository* in the Privacy Agent.

(1)R_{new}.data $\subset$ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition = R_{old}.contextcondition
R_{new}.behvaior = R_{old}.behavior

(2)R_{new}.data $\supset$ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition = R_{old}.contextcondition
R_{new}.behvaior = R_{old}.behavior

(3)R_{new}.data = R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition $\subset$ R_{old}.contextcondition
R_{new}.behvaior = R_{old}.behavior

(4)R_{new}.data = R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition $\supset$ R_{old}.contextcondition
R_{new}.behvaior = R_{old}.behavior

(5)R_{new}.data $\subset$ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition $\subset$ R_{old}.contextcondition
R_{new}.behvaior = R_{old}.behavior

(6)R_{new}.data $\supset$ R_{old}.data
R_{new}.policycondition = R_{old}.policycondition
R_{new}.contextcondition $\supset$ R_{old}.contextcondition
R_{new}.behvaior = R_{old}.behavior

Figure 6.13 *Redundancy profile*

As illustrated in both redundancy and conflict profiles, policy conditions of two preference rules in every situation are always equal. This makes sense as preference rules having different policy conditions apply to different situations and are thus not comparable. Also shown in the conflict and redundancy profiles is that detecting preference conflicts and redundancy involves semantic analysis of privacy data elements and disclosure conditions. This, however, could not be easily (if not impossible) conducted by traditional programming methods based on purely syntactic analysis.

The author have developed a full set of 271 inference rules to describe various relations specified in the Privacy Preference Rule Ontology, so as to deal with semantic reasoning of all potential conflict and redundancy defined in the above profiles. The inference rules are written using HP's Jana Generic Rule Expression, and are reasoned based on the Jena Generic Rule Engine [141]. The Jena Generic Rule Engine is chosen for the proof-of-concept implementation of the *Preference Evaluator*, a key

component in the Privacy Agent, as it supports rule-based inference over OWL/RDF graphs and provides forward chaining, backward chaining, and a hybrid execution model [141]. It provides an approach to overcome an important limitation of using OWL language. The limitation is that the OWL specification currently does not provide direct support for axiomatic rules over ontology description, and thus limits the types of reasoning that are possible with OWL alone.

The author has followed two steps to develop the 271 inference rules:

- First, using Jena Generic Rule Expression to represent each type of preference conflict and redundancy that are illustrated in Figure 6.12 and Figure 6.13. The representation of the conflict and redundancy profile is listed in the first two parts of Appendix E.
- Second, using Jena Generic Rule Expression to represent all relations that are used in conflict and redundancy detection process. These relations are indeed defined as properties in the Privacy Preference Rule Ontology. They are equalTo_p, equalTo_c, contain_c, containedBy_c, mutuallySubsume_c and mutuallySubsumedBy_c. Each relation is defined by enumerating all possible situations that lead to such a relation. For instance, there are eight situations that two preference rule have equal policy conditions (equalTo_p). The representation of the relation definition is listed in the rest parts of Appendix E.

As an example, Figure 6.14 illustrates inference rules that are written using Jena Generic Rule Expression, and are used to detect Non-Modality Conflict of type (10) and redundancy of type (6). Every inference rule consists of multiple first-order predicts, each of which is structured using a triple (subject, verb, object). Subjects and objects with prefix "?" are variables. Verbs represent relations linking subjects and objects. The prefix "pppl" refers to the Privacy Preference Rule Ontology, while the prefix "rdf" and "owl" refer to the OWL specification. An inference rule may also contain Jena built-in primitives, such as notEqual (?x ?y) and equal(?x ?y). The author presents in

Appendix E the most common inference rules that are used to detect preference conflict and redundancy. The author has conducted a series of experiments to evaluate the performance of semantic reasoning based on the 271 inference rules specified, this will be presented in Chapter 7.

```
[NonModalityConflict10:
(?rulex pppl:nonModalityConflict ?ruley) ←
(?rulex rdf:type pppl:Rule)
(?ruley rdf:type pppl:Rule)
(?rulex pppl:hasBehavior ?bx)
(?ruley pppl:hasBehavior ?by)
notEqual(?bx ?by)
(?rulex pppl:hasData ?dx)
(?ruley pppl:hasData ?dy)
(?dx rdfs:subClassOf ?dy)
(?rulex pppl:hasPolicyCon ?pcx)
(?ruley pppl:hasPolicyCon ?pcy)
(?pcx pppl:equalTo_p ?pcy)
(?rulex pppl:hasContextCon ?ccx)
(?ruley pppl:hasContextCon ?ccy)
(?ccx pppl:mutuallySubsumedBy_c ?ccy)
]
```

```
[RedundancyDetecting6:
(?rulex pppl:redundancyWith ?ruley) <-
(?rulex rdf:type pppl:Rule)
(?ruley rdf:type pppl:Rule)
(?rulex pppl:hasBehavior ?bx)
(?ruley pppl:hasBehavior ?by)
equal(?bx ?by)
(?rulex pppl:hasData ?dx)
(?ruley pppl:hasData ?dy)
(?dy rdfs:subClassOf ?dx)
(?rulex pppl:hasPolicyCon ?pcx)
(?ruley pppl:hasPolicyCon ?pcy)
(?pcx pppl:equalTo_p ?pcy)
(?rulex pppl:hasContextCon ?ccx)
(?ruley pppl:hasContextCon ?ccy)
(?ccy pppl:containedBy_c ?ccx)
]
```

Figure 6.14 *Inference rules that are used to detect Non-Modality Conflict of type (10) and redundancy of type (6) using Jena Generic Rule Expression*

6.4.2 Resolving Preference Conflicts and Redundancy

Various conflicting and redundancy situations require different methods to achieve their resolution. The aim of conflict resolution is to ensure that at most one rule is applicable to certain data under certain disclosure condition, while the goal of redundancy resolution is to ensure that no duplicate preference rules exist in the *Privacy Preferences Repository.*

According to pioneering research work on conflict resolution by Dunlop el al. [142], a practical method of resolving potential conflicts is to identify which policy involved in a conflict situation will take precedence. In the author's work, Non-Modality conflicts

involved in privacy preferences are resolved by establishing precedence of two potentially conflicting preference rules. Note that the approach does not attempt to chain the precedence of multiple (i.e. more than two) preference rules. In plain text, if Rule1 have precedence over Rule 2, and Rule2 have precedence over Rule 3, it does not take it for granted to conclude that Rule 1 has precedence over Rule 3. This is because Rule 1 and Rule 3 may not have the same relevancy in terms of data and/or condition elements to rule 2. As a result, Rule1 and Rule 3 are irrelevant and incomparable. The author applies the following two methods to establish the precedence in Non-Modality conflict situations.

- *Specific overrides general:* A preference rule is more specific than another when its application scope (in terms of its data elements and/or condition elements) covers that of another preference rule. In the case exemplified previously, the forbidden rule specified for home-contact information has precedence over the more generic permit rule that is applied to contact information. In fact, allowing two such conflicting rules to coexist has the meaning that the home-contact information is forbidden while all other types of contact information (e.g. business contact, school contact, etc.) are allowed. Since attribute values of data and condition elements of privacy preference rules are structured hierarchically using ontological modeling techniques, the specificality relationship can be easily captured.
- *Assigning explicit priorities to rules:* as illustrated in the Privacy Preference Rule Ontology (Figure 6.1), an atomic preference rule is optionally associated with hasPrecedenceOver property that is specified to establish precedence of two preferences rules. The ontology also places limits so that every atomic preference rule has at most one hasPrecedenceOver property (i.e. maxCardinality of the hasPrecedenceOver property =1). Since meaningful priorities are notoriously difficult for users to assign and it may result in arbitrary priorities that do not really relate to the importance of the policies [140], this method is employed only when a potential Non-Modality conflict is detected. In other words, it does not let users specify a priority when they edit their privacy preferences. Conflict analysis

conduced by the author shows that the method reduces the complexity of the detection and resolution process.

The two precedence establishment methods are manipulated in order. The "*Specific overrides General*" method will be employed by default by the *Preference Evaluator* in the Privacy Agent to resolve no-Modality Conflict. In case that the Privacy Agent cannot determine which preference rule is more specific than another, and thus should take precedence in conflict cases, the precedence will be explicitly established by human users. This indecisive situation happens when multiple subsumption relations exist in data elements and disclosure conditions of two preference rules. A concrete example is that one preference rule is specified as "no information is disclosed to someone I do not know", while another stipulates that "my professional information is allowed to be disclosed to anyone". In this case, two preference rules with different rule behaviors have reverse subsumption relationship applied to data elements and disclosure conditions. As a result, neither rule can be deemed to be more specific than the other.

Compared to the resolution of Non-Modality conflicts, approaches to resolve Modality Conflict and Redundancy are more straightforward. Modality conflicts can be resolved by asking for a person's consent to discard one of the conflict rules, while resolving redundancy is conducted by the Privacy Agent to apply the *specific overrides general* method, i.e. discarding the more specific one of two preference rules, as its application scope is covered by that of another. Evaluation work conducted in Chapter 7 proves that the resolution approaches taken by the author are sufficient to resolve all potential conflicts and redundancy situations of privacy preferences.

6.5 Using Semantic Reasoning to Conduct Privacy Policy Evaluation

In addition to detecting preference conflict and redundancy, ontology-based semantic reasoning is also employed to evaluate data collecting policies against users' privacy preference rules. The author refers to the process of evaluating data collecting policies based on individual privacy preference as policy evaluation. A key step in the policy evaluation process is to select semantically matched preference rules that are stored in the Privacy Preferences Repository and can be applied to evaluate data collecting policies. This section presents how semantic reasoning over an ontology description of the Privacy Preference Rule Ontology can be employed to select appropriate evaluation rules, in addition to describing the policy evaluation process.

6.5.1 Semantic reasoning to select fully-matched and partially matched preference rules

They are broadly two types of preference rules that may be selected to evaluate data collecting policies: fully-matched and partially-matched. As their names suggest, a preference rule is fully matched to evaluate data collecting policies when its application scope (in terms of data elements and disclosure conditions) fully covers that of a policy rule that comprises the data collecting policies, while a partially-matched preference rule is only able to partially cover the application scope of a policy rule, and it must be used together with the specification of a Privacy Meta-Policy when evaluating the policy rule[18].

[18] Like privacy preferences, data collecting policies are transformed into multiple policy rules before they can be evaluated by the *Policy Evaluator* in the Privacy Agent. The transforming procedure will be discussed in the next section.

Fully matched (4)

(1)R_{policy}.data = $R_{preference}$.data
R_{policy}.policycondition $\subseteq$ $R_{preference}$.policycondition
R_{policy}.target = $R_{preference}$.target

(2)R_{policy}.data $\subset$ $R_{preference}$.data
R_{policy}.policycondition $\subseteq$ $R_{preference}$.policycondition
R_{policy}.target = $R_{preference}$.target

(3)R_{policy}.data = $R_{preference}$.data
R_{policy}.policycondition $\subseteq$ $R_{preference}$.policycondition
R_{new}.target $\subset$ $R_{preference}$.target

(4)R_{policy}.data $\subset$ $R_{preference}$.data
R_{policy}.policycondition $\subseteq$ $R_{preference}$.policycondition
R_{policy}.target $\subset$ $R_{preference}$.target

Partially matched (4)

(1)R_{policy}.data $\supset$ $R_{preference}$.data
R_{policy}.policycondition $\subseteq R_{preference}$.policycondition
R_{policy}.target = $R_{preference}$.target

(2)R_{policy}.data = $R_{preference}$.data
R_{policy}.policycondition $\subseteq R_{preference}$.policycondition
R_{policy}.target $\supset$ R_{old}.target

(3)R_{policy}.data $\supset$ $R_{preference}$.data
R_{policy}.policycondition $\subseteq R_{preference}$.policycondition
R_{policy}.target $\supset$ $R_{preference}$.target

(4)R_{policy}.data $\supset$ $R_{preference}$.data
R_{policy}.policycondition $\subseteq R_{preference}$.policycondition
R_{policy}.target $\subset$ $R_{preference}$.target

Figure 6.15 ***Fully-matched and partially-matched preference rule profiles***

Figure 6.15 summaries four situations that could lead to finding fully-matched preference rules, and four situations that could lead to finding partially-matched preference rules. R_{policy} stands for an atomic policy rule that is transformed from the data collecting policy, while $R_{preference}$ is an atomic preference rule stored in the *Privacy Preferences Repository* in the Privacy Agent. The "$\subset$" and "$\supset$" symbols represent subsume_d and subsumedBy_d relations of data elements, or contain_c and containedBy_c relations of context conditions as defined in the Privacy Preference Rule Ontology in section 6.2. For instance, the "R_{policy}.data $\subset$ $R_{preferenc}$.data" expression means that the data value of a policy rule is a subset of that of a preference rule, while the expression R_{policy}.target $\supset$ $R_{preference}$.target" implies that target condition of a preference rule is contained by that of a corresponding policy rule. The expression "R_{policy}.policycondition $\subseteq R_{preference}$.policycondition" states that policy condition of a policy rule is equal or contained by that of a corresponding preference rule. The policy condition are constrains on Purpose, Retention, and Recipient, according to the specification of the Privacy Policy/Preference Language.

It is noted in both fully-matched and partially-matched profiles that only Target

elements of the context condition are compared. This is because data collecting policies stated by data requesters are different from individual privacy preferences. They do not specify constrains on their data collecting practices pointing to a context condition like a user's location, activities and time. Also shown in the fully-matched and partially-matched profiles is that selecting appropriate preference rules for policy evaluation involves semantic reasoning over privacy data elements and disclosure conditions. This, however, could not be easily (if not impossible) conducted by traditional programming methods based on purely syntactic analysis.

The author has developed a full set of 55 inference rules to describe various relationships specified in the Privacy Preference Rule Ontology, so as to deal with semantic reasoning to find fully-matching and partially-matching preference rules. Among the 55 inference rules, 35 inference rules are used to conduct reasoning to find fully-matched applicable preference rules, while 47 inference rules are used to find partially-matched applicable preference rules. They share 27 inference rules that define equalTo_p, contain_p and containedBy_p relations. Like inference rules presented in the last section for conflict and redundancy detection, these inference rules are written using HP's Jana Generic Rule Expression, and are reasoned based on the Jena Generic Rule Engine.

The author has followed two steps to develop the 55 inference rules:

- First, using Jena Generic Rule Expression to represent each type of fully-matched and partially-matched selection that is illustrated in Figure 6.15. The representation of the fully-matched and partially-matched selection is listed in the first two parts of Appendix F.
- Second, using Jena Generic Rule Expression to represent all relations that are used in policy evaluation process. These relations are indeed defined as properties in the Privacy Preference Rule Ontology. They are equalTo_p, contain_p, and containedBy_p. Each relation is defined by enumerating all possible situations that lead to such a relation. For instance, there are twenty-seven situations that two

preference rule have equal or contained policy conditions (equalTo_p or contained_p, or containedBy_p). The representation of the relation definition is listed in the rest parts of Appendix F.

```
[fullyMatchRule2a:
(?rulex pppl:fullyMatch ?ruley) ←
(?rulex rdf:type pppl:Rule)
(?ruley rdf:type pppl:Rule)
(?rulex pppl:hasData ?dx)
(?ruley pppl:hasData ?dy)
(?dx rdfs:subClassOf ?dy)
(?rulex pppl:hasPolicyCon ?pcx)
(?ruley pppl:hasPolicyCon ?pcy)
(?pcx pppl:equalToOrContainedBy_p ?pcy)
(?rulex pppl:hasContextCon ?ccx)
(?ccx pppl:hasTarget ?tax)
(?ruley pppl:hasContextCon ?ccy)
(?ccy pppl:hasTarget ?tay)
equal(?tax ?tay)
]
```

```
[fullyMatchRule2b:
(?rulex pppl:fullyMatch ?ruley) ←
(?rulex rdf:type pppl:Rule)
(?ruley rdf:type pppl:Rule)
(?rulex pppl:hasData ?dx)
(?ruley pppl:hasData ?dy)
(?dx rdfs:subClassOf ?dy)
(?rulex pppl:hasPolicyCon ?pcx)
(?ruley pppl:hasPolicyCon ?pcy)
(?pcx pppl:equalToOrContainedBy_p ?pcy)
(?rulex pppl:hasContextCon ?ccx)
noValue(?ccx pppl:hasTarget ?tax)
(?ruley pppl:hasContextCon ?ccy)
noValue(?ccy pppl:hasTarget ?tay)
]
```

Figure 6.16 *Inference rules to select fully matched preference rule of type (2)*

```
[partiallyMatchRule1a:
(?rulex pppl:partiallyMatch ?ruley) ←
(?preference rdf:type pppl:Preferences)
(?preference pppl:hasMetaPolicy ?metapolicy)
equal(?metapolicy pppl:Optimistic)
(?rulex rdf:type pppl:Rule)
(?ruley rdf:type pppl:Rule)
(?ruley pppl:hasBehavior ?by)
equal(?by pppl:Permit)
(?rulex pppl:hasData ?dx)
(?ruley pppl:hasData ?dy)
(?dy rdfs:subClassOf ?dx)
(?rulex pppl:hasPolicyCon ?pcx)
(?ruley pppl:hasPolicyCon ?pcy)
(?pcx pppl:equalToOrContainedBy_p ?pcy)
(?rulex pppl:hasContextCon ?ccx)
(?ccx pppl:hasTarget ?tax)
(?ruley pppl:hasContextCon ?ccy)
(?ccy pppl:hasTarget ?tay)
equal(?tax ?tay)
]
```

```
[partiallyMatchRule1b:
(?rulex pppl:partiallyMatch ?ruley) ←
(?preference rdf:type pppl:Preferences)
(?preference pppl:hasMetaPolicy ?metapolicy)
equal(?metapolicy pppl:Pessimistic)
(?rulex rdf:type pppl:Rule)
(?ruley rdf:type pppl:Rule)
(?ruley pppl:hasBehavior ?by)
equal(?by pppl:Forbid)
(?rulex pppl:hasData ?dx)
(?ruley pppl:hasData ?dy)
(?dy rdfs:subClassOf ?dx)
(?rulex pppl:hasPolicyCon ?pcx)
(?ruley pppl:hasPolicyCon ?pcy)
(?pcx pppl:equalToOrContainedBy_p ?pcy)
(?rulex pppl:hasContextCon ?ccx)
(?ccx pppl:hasTarget ?tax)
(?ruley pppl:hasContextCon ?ccy)
(?ccy pppl:hasTarget ?tay)
equal(?tax ?tay)
]
```

Figure 6.17 *Inference rules to select partially matched preference rule of type (1)*

As an example, Figure 6.16 and 6.17 above illustrate respectively inference rules that are written using Jena Generic Rule Expression, and used to select fully-matched rule of type (2) and partially-matched rule of type (1). In Figure 6.16, fullymatchRule2a and fullymatchedRule2b represent two possible situations that lead to a target condition of a preference rule being equal to that of a policy rule. It is illustrated in Figure 6.17 that, partially-matched preference rules are only able to cover partially the application scope of data collecting policy rules, and they must be used together with the specification of Privacy Meta-Policy when evaluating data collecting policies. The author presents in Appendix F a full list of 55 inference rules that are used in the policy evaluation process. The author has conducted a series of experiments to evaluate the performance of semantic reasoning based on the 55 inference rules specified, this will be presented in Chapter 7.

6.5.2. Policy Evaluation Process

The policy evaluation process is conducted following the algorithm illustrated in Figure 6.18. Once the Privacy Agent receives data collecting policies, it reads them into the *Policy Evaluator.* The *Policy Evaluator* is responsible for transforming the data collecting policies into multiple policy rules based on the specification of the Privacy Preference Rule Ontology, and comparing each policy rules with privacy preference rules that are stored in the *Privacy Preferences Repository.* To evaluate each policy rule, the *Policy Evaluator* starts by selecting preference rules that are fully matched to evaluate the policy rule. It looks into partially-matched preference rules when no fully-matched preference rules can be selected. In the case that no matched preference rules (either fully-matched or partially-matched) can be found, the *Policy Evaluator* moves on to evaluate the next policy rule. In contrast, if multiple preference rules can be selected, the *Policy Evaluator* attempts to establish precedence of these matched rules. This leads to either one or null preference rule applicable to evaluate the target policy rule. If such an applicable preference rule can be selected, the *Policy Evaluator*

then looks into the context condition (Target, Location, Time and Activity, if any) of the preference rule, and checks it with a person's actual context. Rule behavior of the matched preference rule will decide whether or not the target policy rule complies with the person's privacy preferences, if no conflict is detected during the context check. If eventually no matched preference rule can be found after going through the policy evaluation process, the Privacy Agent needs to ask for the person's explicit consent for information disclosure decisions.

```
Read (DataCollectingPolicy.xml);
Ruleset[]=TransformToRules (DataCollectingPolicy.xml)
action="AskMe"
For (; Ruleset[].hasnext ; )
{   matchedRuleset[]= FullyMatch(); //To find fully matched preference rules
    If (matchedRuleset[]= null)
     Then {matchedRuleset[]=PartiallyMatch(); //To find partially matched preference rules
              If (matchedRuleSet[]=null)
              Then {move to next policy rule}
            }
     //Comparing priority if multiple applicable preference rules exist
     If (matchedRuleset[].hasNext != null)
       Then { result1=FindRuleWithHigherPriority(matchedRuleset[])
              If (result1 =null)
               Then {action= "AskMe"}
               Else {matchedRule= result1
                     //Check context condition of matched rule against a person's context
                      result2=CheckContext (matchedRule.ContextConditon);
                      If (result2 = True)   //Context condition is satisfied
                        Then {action=matchedRule.behavior}
                        Else {action="Askme"}
                     }
            }
    If (action = "AskMe")
      Then {AskUserConsent; quit the policy evaluation}
      Else {move to the next policy rule}
}
```

Figure 6.18 *Pseudocode of the policy evaluation algorithm*

Like user-defined privacy preferences, data collecting policies received from data context-aware applications need to be manipulated to a format that can be used in the

policy evaluation process. The author has developed four transformation principles[19] to transform data collecting policies received from the data requestor to policy rules that comply with the specification description of the Privacy Preference Rule Ontology. As an example, the *Who-near-me* application's data collecting policy (see Figure 5.1 in Chapter 5) is transformed into 12 policy rules.

```
<rdf:Description rdf:about="#policyrule1_whonearme">
        <rdf:type>
            <rdf:Description rdf:about="#Rule"/>
        </rdf:type>
        <pppl:hasContextCon>
            <rdf:Description rdf:about="#contextcondition1_whonearme"/>
        </pppl:hasContextCon>
        <pppl:hasPolicyCon>
            <rdf:Description rdf:about="#policycondition1_whonearme"/>
        </pppl:hasPolicyCon>
        <pppl:hasBehavior>
            <rdf:Description rdf:about="#Permit"/>
        </pppl:hasBehavior>
        <pppl:hasData>
            <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#realname"/
        </pppl:hasData>
    </rdf:Description>

<rdf:Description rdf:about="#contextcondition1_whonearme">
        <rdf:type>
            <rdf:Description rdf:about="#ContextCondition"/>
        </rdf:type>
        <pppl:hasTarget>
            <rdf:Description rdf:about="http://www.ee.ucl.ac.uk/~jezhang/personalinformationontology#friend"/>
        </pppl:hasTarget>
    </rdf:Description>

<rdf:Description rdf:about="#policycondition1_whonearme">
        <pppl:hasPurpose>
            <rdf:Description rdf:about="#navigation"/>
        </pppl:hasPurpose>
        <pppl:hasRecipient>
            <rdf:Description rdf:about="#ours"/>
        </pppl:hasRecipient>
        <pppl:hasRetention>
            <rdf:Description rdf:about="#stated-purpose"/>
        </pppl:hasRetention>
        <rdf:type>
            <rdf:Description rdf:about="#PolicyCondition"/>
        </rdf:type>
    </rdf:Description>
```

Figure 6.19 *RDF/XML representation of a policy rule transformed from the data collecting policy of the Who-near-me application*

[19] The four transformation principles will be presented in the proof-of-concept implementation in Chapter 7.

Figure 6.19 above illustrates one of these policy rules represented in the RDF/XML scheme. The transformation approach helps maintain the flexibility and good expressiveness of data collecting policies, while at the same time addressing a key problem. The problem is that the ontology-based description logic is inept at dealing with a logic operand like "or" and "and".

6.6 Use Semantic Reasoning to Perceive Contextual Knowledge

In order to make the appropriate information disclosure decision, the Privacy Agent needs to perceive its owner's context. In section 6.3, the author has followed a formal ontology-based approach to represent four staple categories of context information: Personal Information, Location, Time, and Activity. This section presents how hybrid reasoning mechanisms can be conducted over the Context Ontology to perceive a person' context.

Table 6.4 *Fact-based contexts*

	Intrinsic	**Extrinsic**
Static	Personal profile (age, gender, etc) Business contact (email, workplace, employer, etc) Social relationships (family members, colleagues, alumni, etc) Physical environment (city, district, street, etc)	A room is part of a building. Sister belongs to family members.
Dynamic	Location, Time	A person is in a specific location.

In addition to classifying context information into four staple categories, here context information is seen to consist of two basic types: fact-based context and situation-based context. The fact-based context, as its name suggests, captures basic facts about an

entity (e.g. a person, a place, etc) and its relationships with other entities. As shown in Table 6.4 above, a common feature of fact-based contexts is that they can be directly gathered through interacting with sensors or by inputting manually by human users. In contrast, the situation-based context represents a higher-level abstract description of real-world situations, such as "a person is in a meeting". This cannot be obtained directly from sensing environments, and is normally derived from combining various individual facts. In this work, the author refers the process of deriving higher-level situation abstraction of context from the low-level context fact as *Context Reasoning*. Note that the process is not concerned with context acquisition mechanisms themselves, but assumes that context information can be provided by appropriate context providers whenever needed.

Two types of inference logics are exploited in the context reasoning process: Ontology-based description logic and rule-based application-specific logic. The ontology-based description logic includes logical constructors such as the intersection or conjunction of concepts, union or disjunction of concepts, negation or complement of concepts, value restriction, existential restriction, etc. Other constructors may also include restrictions on relations, for example, inverse, transitivity, functionality, etc. These constructors allow reasoning mechanisms to deduce further knowledge from the original ontology description. For instance, Description Logic can be used to support inferences that a spatial entity "Oxford Street" is geographically located in another spatial entity "UK", as by the Description Logic definition "Oxford Street" is located in "London" and "London" is in the "UK". The rule-based application-specific logic is applied to derive new relationships from existing ones based on description logic constructs, and to allow situation-based contexts and dynamic extrinsic contexts to be derived. A simple example is that, knowing a person "Alice" is currently located in Oxford Street, which by description logic definition is in turn in London, the application-specific logic can draw conclusion that Alice is located in London.

The following parts present the application of the hybrid reasoning capabilities that

combines ontology-based description logic and rule-based application-specific logic in spatial, temporal, and activity reasoning respectively.

- ***Spatial Reasoning***

Location Ontology developed in this work captures spatial concepts and relations that are used by human users when specifying their privacy preferences, and by Privacy Agent to perceive properly users' context when enforcing the privacy preferences. This work denotes the logical inference of reasoning over the spatial information of a person's location context as *spatial reasoning*. The process of spatial reasoning may involve mapping between spatial concepts having different formats and representations (e.g. mapping location coordinate data to symbolic description of the geographic address), correlating the location information of various spatial entities (such as a restaurant, a person's current location and a university), and deriving new spatial relations from existing ones, for example, knowing A is located in B, B is geographically disconnected from C, it then infers that A is geographically disconnected from C.

The spatial reasoning over Location Ontology can be used by a Privacy Agent in various ways. First of all, the Privacy Agent can use spatial reasoning to enforce ambiguous disclosure decisions. Since the Location Ontology structures various spatial concepts in a hierarchical way, the spatial reasoning conducted over the Location Ontology can protect a person's privacy by adjusting the granularity of his/her location information. Second, it can use spatial reasoning to infer a location's spatial properties that cannot be directly acquired from physical sensors. For example, from the readings of a GPS-enabled device that a person carries, the Privacy Agent can acquire some basic information about the person's location (e.g. the latitude and longitude coordinates). By finding the corresponding symbolic representation of the coordinates in a pre-defined ontology, the Privacy Agent can infer additional spatial information (e.g., the city in which the person is located, or a nearby building that is spatially subsumed by the

university campus in which the person is located). Third, spatial reasoning can be used to detect inconsistent information about a person's location. For example, according to the Location ontology, room A is disconnected from room B, and there are two different reports about the same person being located in both room A and room B during the same time interval. Using spatial reasoning, the Privacy Agent infers a person cannot be simultaneously located in two disconnected rooms. Hence, it concludes the reported location information is inconsistent.

The spatial reasoning is conducted based on various spatial relations defined in the Location Ontology through five properties[20], i.e. *relatedTo*, *locatedIn*, *disconnectedWith*, *spatiallySubsume* and *spatiallySubsumedBy*; in addition to the hierarchical representation of spatial concepts that is also modeled on the Location Ontology. The author implemented a spatial reasoner based on HP Jena API's Generic Rule Reasoner. The Jena Generic Rule Reasoner supports rule-based inference over RDF graphs and provides forward chaining, backward chaining, and a hybrid execution model [141]. When using the Jena API to process OWL data, all RDF statements are stored in an abstract data structure called *model*. The model of the RDF statements can answer queries about the underlying RDF graph. A Generic Rule Reasoner can be attached to a model to provide additional inference support. Figure 6.20 exemplifies some of the inference rules that are written using the Jena Generic Rule Expression, and used to define the spatial reasoner. A full list of spatial reasoning inference rules used in the author's simulation and evaluation work is presented in Appendix G.

[20] A summary of the Description Logic restrictions on these properties is presented in Table 6.2.

- Ontology-based reasoning based on description logic constrains, in particular, *TransitiveProperty*

```
[SpatialReasoning1:
 (?x loc:relatedTo ?z) ←
   (?x rdf:type loc:NetworkAddress)
   (?y rdf:type loc:Coordinate)
   (?z rdf:type loc:GeographicalAddress),
   (?x loc:relatedTo ?y)
   (?y loc:relatedTo ?z)
   ]
```

```
[SpatialReasoning2:
 (?x loc:spatiallySubsumedBy ?z) ←
   (?x rdf:type loc:GeographicalAddress)
   (?y rdf:type loc:GeographicalAddress)
   (?z rdf:type loc:GeographicalAddress)
   (?x loc:spatiallySubsumedBy ?y)
   (?y loc:spatiallySubsumedBy ?z)
   ]
```

- Application-specific logics to define locatedIn and disconnectedWith relations.

```
[SpatialReasoning3:
 (?x loc:locatedIn ?y) ←
   (?x rdf:type loc:ContextualLocation)
   (?y rdf:type loc:PremiseLocation)
   (?x loc:hasGeographicAddress ?x1)
   (?y loc:hasCooridinate ?y1)
   (?x1 loc:relatedTo ?y1)
   ]
```

```
[SpatialReasoning4:
 (?x loc:disconnectedWith ?y) ←
   (?x rdf:type loc:ContextualLocation)
   (?y rdf:type loc:PremiseLocation)
   (?x loc:hasGeographicAddress ?x1)
   (?y loc:hasCooridinate ?y1)
   noValue(?x1 loc:relatedTo ?y1)
   ]
```

Figure 6.20 *Example inference rules used to configure Jena Generic Rule Reasoner to support spatial reasoning*

- ***Temporal Reasoning***

Time Ontology developed in this work capture spatial concepts and relations that are used by human users when specifying their privacy preferences, and by the Privacy Agent to perceive properly users' context when enforcing the privacy preferences. This work denotes the logical inference of temporal order of various temporal concepts defined in the Time Ontology as *temporal reasoning*. The process of temporal reasoning may involve mapping between abstract temporal concepts (e.g. working hours and weekend) and exact time description using seconds, minutes, hours, days, etc, correlating temporal information of various temporal entities (such as today and scheduled date of an activity), and deriving new spatial relations from existing ones, for example, knowing an instant entity A is inside an interval entity B, and B is temporally covered by another interval entity C, it then infers that A is inside C.

The temporal reasoning over the Time Ontology can be used by a Privacy Agent in

various ways. First, temporal information can help the Privacy Agent correlate distinctive contextual information to infer a person's activity. For example, knowing a traveling plan is scheduled for today, and the current location of a person is not in the area of her hometown, the Privacy Agent could infer that the person is traveling. Second, the temporal reasoning can be used to detect inconsistent contextual knowledge. For example, the privacy agent is informed that a person is currently located at home. Later, a new report indicates that the same person is present at work. The event time intervals described in the two reports overlap. Using this temporal ordering information, the Privacy Agent can detect the person's activity context is inconsistent.

The temporal reasoning is conducted based on various spatial relations defined in the Time Ontology through nine properties[21], i.e. *before, after, inside, intDuring, intContain intOverlap intOverlappedby, intEqual* and *insEqual*, in addition to the hierarchical representation of temporal concepts that is also modeled in the Time Ontology. The author implemented a temporal reasoner based on HP Jena API's Generic Rule Reasoner. Figure 6.21 exemplifies some of the inference rules that are written using the Jena Generic Rule Expression, and used to define the temporal reasoner. A full list of temporal reasoning inference rules used in the author's simulation and evaluation work is presented in Appendix G.

[21] A summary of the Description Logic restrictions on these properties is presented in Table 6.3.

- Ontology-based reasoning based on description logic constrains, in particular, *TransitiveProperty*

```
[SpatialReasoning1:
 (?x tme:before ?z) ←
   (?x rdf:type tme:IntervalEntity)
   (?y rdf:type tme:InstantEntity)
   (?z rdf:type tme:IntervalEntity)
   (?x tme:before ?y)
   (?y tme:before ?z)
   ]
```

```
[TemporalReasoning2:
 (?x tme:inside ?z) ←
   (?x rdf:type tme:InstantEntity)
   (?y rdf:type tme:IntervalEntity)
   (?z rdf:type tme:IntervalEntity)
   (?x tme:inside ?y)
   (?y tme:intDuring ?z)
   ]
```

- Application-specific logics to define intEqual and intOverlap relations.

```
[TemporalReasoning3:
 (?x tme:intEqual ?y) ←
   (?x rdf:type tme:IntervalEntity)
   (?y rdf:type tme:IntervalEntity)
   (?x tme:begins ?beingsX)
   (?x tme:ends ?endsX)
   (?y tme:begins ?beginsY)
   (?y tme:ends ?endsY)
    equal(?beginsX ?beginsY)
    equal(?endsX ?endxY)
   ]
```

```
[TemporalReasoning4:
 (?x tme:intOverlap ?y) ←
   (?x rdf:type tme:IntervalEntity)
   (?y rdf:type tme:IntervalEntity)
   (?x tme:begins ?beginsX)
   (?x tme:ends ?endsX)
   (?y tme:begins ?beginsY)
   (?y tme:ends ?endsY)
   lessThan(?beginsX ?beginsY)
   greaterThan(?endsX ?beginsY)
   lessThan(?endsX ?endsY)
   ]
```

Figure 6.21 *Example rules used to configure Jena Generic Rule Reasoner to support temporal reasoning*

- ***Activity Reasoning***

Human activities are embraced by situation-based context and happen in certain time and location. Perceiving a person's activities thus involves the combination of spatial reasoning, temporal reasoning, and the retrieval of personal information. The process of activity reasoning requires deriving high-level situation-based contexts from low-level context facts regarding location and time. Unlike spatial and temporal reasoning, in which the inference process is partially based on ontology-based description logic, most of the activity reasoning is conducted using rule-based application-specific logic. There exists various ways to perceive human activities. Inference rules can be very complicated based on Artificial Intelligence and machine learning techniques, or can be simply specified according to common sense. In this work, the author demonstrates

some simple inference rules based on common sense to show how the rule-based application-specific logic over ontology description works.

Like spatial and temporal reasoners presented previously, Jena API's Generic Rule Reasoner is chosen to implement the activity reasoner. Figure 6.22 presents two simple inference rules that are written in Jena's Generic Rule Expression and used in the author's simulation and evaluation work.

Inference Rule 1: *If Alice's current location is inside her working place and current time is within working hours, it can be inferred that she is working.*

Inference Rule 2: *If Alice's current location is not in the area of her hometown, and the schedule in her diary shows she may be in traveling today, it can be inferred that she is traveling.*

```
[InferenceRule1:
 (?p per:isEngagedIn act:working) ←
   (?p rdf:type per:person)
   (?p per:hasContextualLocation ?lp)
   (?lp rdf:type loc:ContextualLocation)
   (?p per:hasWorkingPlace ?com)
   (?com rdf:type loc:SpatialEntity)
   (?com loc:hasLocation ?lc)
   (?lc rdf:type loc:PremiseLocation)
   (?lp loc:locatedIn ?lc) ←
       (?lp loc:hasGeograhicAddress ?addressA)
       (?lc loc:hasGeograhicAddress ?addressB)
       (?addressA rdf:type loc:GeographicalAddress)
       (?addressB rdf:type loc:GeographicalAddress)
       (?addressA loc:relatedTo ?addressB)
 ]
```

```
[InferenceRule2:
   (?p per:isEngagedIn act:traveling) ←
    (?p rdf:type per:person)
    (?p per:hasContextualLocation ?lp)
    (?lp rdf:type loc:ContextualLocation)
    (?p per:hasHome ?home)
    (?home rdf:type loc:SpatialEntity)
    (?home loc:hasLocation ?lh)
    (?lh rdf:type loc:PremiseLocation)
    (?lp loc:disconnectedWith ?lh) ←
        (?lp loc:hasGeograhicAddress ?addressA)
        (?lp loc:hasGeograhicAddress ?addressB)
        (?addressA rdf:type loc:GeographicalAddress)
        (?addressB rdf:type loc:GeographicalAddress)
        noValue (?addressA loc:relatedTo ?addressB)
    (?p per:hasScheduledActivity ?act)
    equal(?act act:traveling)
    (?act act:HappenWhen ?dates)
    (?dates tme:intContain tme:today)
    ]
```

Figure 6.22 *Example rules used to configure HP Generic Rule Reasoner to support activity reasoning*

6.7 Related Work

- ***Using Ontology-based Methods for Policy Representation and Reasoning***

The use of rule-based policy is common in computing systems that feature security or privacy protection [143]. Increasing interest in Semantic Web and ontologies in the last few years has led to emerging approaches that exploit Semantic Web languages to represent policies, in particular, Rei[22] [144], KAoS [145], SOUPA [103], and Semantic e-Wallet [17] are among a few salient attempts. KAoS uses DAML (and appears to have moved to using OWL[23]) as the basis for representing and reasoning about policies within Web Services, Grid Computing, and multi-agent system platforms. KAoS exploits ontologies for representing and reasoning about *domains* describing organizations of human, agents, and other computational actors. Rei is a logic-based policy language that was originally grounded in a semantic representation of policies using RDF-S, and its recent development has moved to an OWL implementation[24]. The SOUPA policy language is represented using OWL, and is similar to Rei in modeling a policy as a set of rules that define restrictions on actions (e.g. an action with certain properties is permitted or forbidden). Semantic e-Wallet is another effort to use a Semantic Web language, in particular OWL, to represent data access policies and to protect individual privacy. Distinctive from the informal way of specifying policies in Semantic e-Wallet, policy languages specified in KAoS, Rei, and SOUPA reuse many policy management concepts developed in Ponder [146], a pioneering object-oriented policy language developed by Imperial College for the management of distributed systems and networks.

The ontology-based solution proposed in the author's work is similar to the above work

[22] Rei, is a Japanese 'Kanji' character which means 'universal' or 'essence'. It was chosen to indicate the universal applicability of the policy language, including security, conversation and management [144].
[23] More information is presented in http://ontology.ihmc.us/ontology.html
[24] More information is presented in http://rei.umbc.edu/

in terms of advocating the use of an ontology-based approach for policy presentation, but it is distinct from them in the specification and use of policy. Policies specified using KAoS, Rei and SOUPA policy language are used to provide access control to resources. Although SOUPA and a further research [147] based on Rei claimed that Rei can be used for express privacy policy, both SOUPA and Kolari's work demonstrated that they are still system-centric attempts to protect privacy by implementing centralized access control on sensitive information within their system. In contrast, the Privacy Policy/Preference Language developed by the author is grounded in an important user-centric privacy practice —— the Platform for Privacy Preference Project (P3P). Policies specified in the above related work are neither similar to data collecting policies proposed in this work that are meant to enhance people trust in disclosing their personal information by providing them background information about the disclosure (i.e. Purpose, Recipient, Retention, etc.), nor having function to support personal privacy preferences in response to various context (i.e. Target, Location, Time and Activity).

Different policy representations lead to different computing approaches to reason and analyze policies. The design of the KAoS policy ontology and SOUPA suggests the use of a description logic inference engine to analyze policy rules, although they use different policy language representation. On the other hand, the Rei policy ontology requires the use of a Frame Logic (F-Logic) based meta-interpreter to compute policy restrictions and constraints. The policy analysis mechanisms in e-Wallet system exploits the XSLT technology [169] to translate policy rules from RDF to JESS rules, and uses a JESS rule engine to compute policy restrictions.

Comparing with the previous efforts, the approaches taken by the author to use hybrid reasoning mechanisms that combine ontology-based Description Logic and rule-based application-specific logic for semantic policy analysis of personal privacy preferences are novel. For one thing, F-Logic-based meta-interpreter in Rei [144] and JESS rule engine in [17] do not support semantic reasoning using ontology-based Description

Logic, they took traditional analysis methods based purely on policy syntax. For another, although KAoS [145] and SOUPA [103] also advocated taking advantage of ontology-based description logic to conduct semantic policy analysis, little detail has been given by them to demonstrate how their methods work, and none of them give experimental evidence to show the performance of their semantic policy analysis. In contrast, the author described in this chapter how the hybrid reasoning mechanisms over ontology-based description can be used for privacy policy evaluation as well as detecting preference conflict and redundancy, and has carried out a series of experiments to evaluate and prove the approaches taken. The evaluation work will be presented in Chapter 7. As a key contribution, the author's experience with ontology-based methods has revealed the potential for taking advantage of semantically-rich policy representation and reasoning to reduce human error, simplify policy analysis, detect policy conflicts, and facilitate policy understanding.

- ***Context Information Modeling and Context Reasoning***

The author's work on modeling context information has benefited from some salient ontology work, in particular, Friends-of-A-Friends ontology (FOAF) [148] that captures online personal information and relationships, DAML-Time [135] and OWL-Time [136] that express temporal concepts, and the spatial ontologies in OpenGIS [137] that symbolically represent spatial objects and relations. In addition to these dedicated pieces of work developed for representing single category of information, ontology-based methods have been employed by context-aware computing research to model various contextual information used in ubiquitous computing environments. The trend reflects the potential of ontology-based approaches to address critical issues including formal context representation, knowledge sharing and logic-based reasoning about context. Some examples of recent work include Context Ontology (CONON) [48] by Wang et al., the Standard Ontology for Ubiquitous and Pervasive Applications (SOUPA) [103] by Chen et al., Context-Aware Middleware for Ubiquitous Computing Systems (CAMUS) by Shehzad et al. [138], a context model by Henricksen et al. [53],

the Agent-Based Context-Aware Infrastructure (ACAI) by Khedr and Karmouch [149], ONTO-CONTEXT by Serrano et al. [150], the work in Ambient Network [139], and the Flow Context by Ocampo[170].

More specifically, the CONON is developed for pervasive computing applications, and is structured into two levels: an upper-level ontology that captures general concepts about context, and lower domain-specific ontologies tailored to specific application areas [91]. Some of the concepts enumerated in CONON's upper-level ontology include Location, Person, Activity and CompEntity. Under CompEntity one may find the subconcepts Service, Application, Device, Network and Agent. The SOUPA is also designed to provide support for pervasive computing applications [103]. It includes vocabularies to express concepts associated with persons, agents, belief-desire-intention (BDI), actions, policies, time, spaces and events. Context modeling work by Henricksen et al. [53] is interesting, since it attempts to incorporate privacy support into their context model by providing a set of ontological concepts to describe information ownership (such as using, owns, permitted to use, and controls) and express privacy preferences (such as prohibit, oblige, etc). Other concepts in its demonstration work include communication channel, device, person, activity, place, organization, etc. The CAMUS architecture provides the following domain concepts in its ontology: Agent, Environment, Device, Location, and Time [138]. The Agent-Based Context-Aware Infrastructure (ACAI) [149] presents ontologies for pervasive environments and describes concepts related to locations, actors, networks, services and actions. The work in Ambient Network [139] includes generic network-related concepts such as Network and WirelessNetwork classes, concepts related to Ambient Network nodes, and even cost and charging. Similar to the focus in the Ambient Networks project, the Flow Context proposed by Ocampo [170] captures network-related concepts; in particular, it encompasses any information that can be used to characterize a sequence of protocol data units within a network. The information is not only about the flow itself (i.e., "what it is transporting") but also about the entities that are relevant to or associated with the flow, such as the users that generate the flows, their activities, their

devices, the applications, and others.

The Context Ontology developed by the author is distinguished from these ontology-based context models in several important ways. First of all, the concepts and relationships captured in the author's Context Ontology are pertinent to human users (in particular, human activities, their location, personal information, and time), and are mainly used to facilitate the delivery of context-aware applications and services. This appears not to be the focus of many ontology-based context models presented above. CAMUS [138] and SOUPA [103] mainly include concepts that describe computing agents, their behaviors and communication among the agents to achieve automation, while in Ambient Network [139], ACAI [149] and Flow Context [170], ontology work focuses on network-related context information. Although the SOUPA [103] and context model [53] by Henricksen et al. include some concepts relevant to human users and their privacy requirements, these concepts are not sufficient to express contextual knowledge that are required in the author's work to express individual privacy preferences in response to various context.

Secondly, most of the above context ontologies (except SOUPA [103]) limited the use of ontology only to represent context information and relationships between information. In contrast, the author has employed ontological modeling approaches to express privacy vocabulary. By taking advantage of the real power of ontology-based semantic reasoning, the Privacy Agent could perceive a person's context properly before judging the acceptability of data requesters' collecting policies and take appropriate actions on behalf of individuals.

Thirdly, unlike many context models, such as CONON[48], CAMUS [138], ACAI [149], Ambient Network [139] and Flow Context [170], which tried to capture as many concepts and relationships as possible in an intended domain, all individual ontologies developed in the author's work only include concepts and relations that are necessary for performing context perception and privacy check. This helps avoid unnecessary

complexity that is more than required by most simple applications, and more than affordable by the hosts of the Privacy Agent that have limited CPU power, memory and computing facility. In addition, among very few attempts, the author has conducted quantitative performance evaluation on semantic reasoning over the context ontologies developed. In contrast, many previous researches on ontology-based context models did not provide experiment-based evidence to validate the feasibility of their models.

6.8 Chapter Summary

This chapter described in detail the author's work on exploiting ontology-based methods to model the Privacy Policy/Preference Language developed in Chapter 5, and experimenting on using hybrid reasoning mechanisms for semantic policy analysis and context reasoning. The approaches taken by the author and reasons behind these were discussed.

Employing ontology-based techniques for policy representation and reasoning is a key part of the proposed Privacy Agent technology. It facilitates automated process of privacy control, and helps achieve the ultimate goal of the author's privacy protection solution —— to deliver relatively unobtrusive privacy management, and to empower people to manage their privacy in dynamic context-aware ubiquitous computing environments with relative ease.

In the following Chapter 7, the author will present the proof-of-concept implementation of some key parts of the proposed privacy protection model and privacy agent technology, and conduct a series of experiments to evaluate the approaches taken by the author for semantic policy analysis and context reasoning. As a key contribution of this doctoral work, the author's experience with ontology-based techniques has revealed their potential of having a semantically rich policy representation and reasoning, to

reduce human error, to simplify policy analysis, to detect policy conflicts, and to facilitate policy understanding.

Chapter 7: A Prototype Implementation and Quantitative Performance Evaluation of Semantic Reasoning

This chapter demonstrates a proof-of-concept implementation of some key parts of the distributed privacy protection model and intelligent agent technology proposed in previous chapters. The implementation focuses on privacy interactions between human users and the Privacy Agent and between the Privacy Agent and context-aware applications. In addition, it focuses on staple functionalities of the Privacy Agent with respect to the privacy policy/preference evaluation and context perception. Since the author's proposal is independent of specific context-aware architectures, systems and middleware solutions, the prototype implementation aims to shed light on how the privacy solution could be employed in real world situations by simulating system environments in which it may be deployed. The chapter also presents a series of experiments to evaluate the performance of the use of hybrid reasoning mechanisms that combine ontology-based Description Logic and rule-based application-specific logic in semantic policy analysis of personal privacy preferences and in context reasoning. The evaluation results contribute to the validation of the novel semantic privacy policy/preference analysis approaches, and to addressing key doubts about the performance of ontology-based methods in resource-constrained environments.

7.1 Proof-of-concept Implementation

In the proof-of-concept implementation, Web Service technologies have been employed to simulate interactions between the Privacy Agent and human users, and between the Privacy Agent and context-aware applications. This technology choice is driven by two major reasons. First, information in the context-aware paradigm is usually derived

from a range of sources and disseminated to diverse applications running on various platforms. The Web Service technologies are suitable to simulate such a heterogeneous environment as they can be developed and used independently of hardware platforms, operating systems and programming languages, and can achieve a certain level of interoperability among diverse devices and applications. Second, ubiquitous computing environments are characterized by the distributed nature of resources. The Web Service technologies, such as a lightweight XML messaging protocol, Simple Object Access Protocol (SOAP) [151], are well suited to exchange information in such decentralized and distributed environments, and they allow application developers to make functionality available across the Internet. In contrast, enterprise distributed technologies, such as Remote Method Invocation (RMI) [152] and Common Object Request Broker Architecture (CORBA) [153], simply do not scale to the global Internet. Because of the advantages of being interoperable, scalable and allowing quick application deployment, the Web and Web Service technologies have been taken up by some salient work [17, 26, 61] in the field of context-aware research for simulation purpose and prototype implementation.

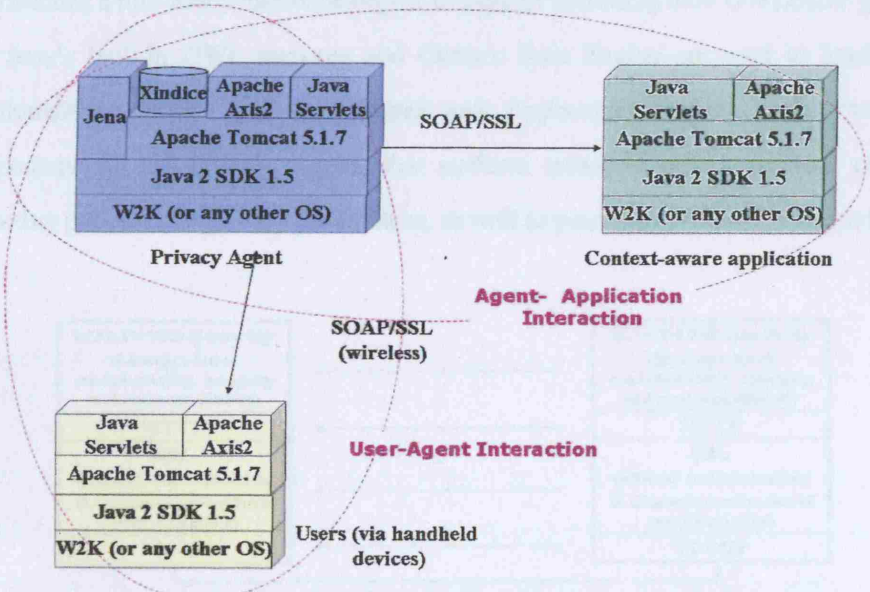

Figure 7.1 *Individual technologies used in the prototype implementation*

Figure 7.1 above presents individual technologies and tools that are employed in the prototype implementation. The Apache Axis2 [154] is an open-source, Java-based implementation of the SOAP 1.2 specification [151], which runs on top of an Apache Tomcat Server [155]. Apache Axis2 supports both SOAP messaging and remote procedure calls via SOAP, i.e. it enables the Tomcat server to support SOAP-RPC interfaces. The Axis2 implementation of SOAP also supports OASIS WS-Security (WS-security) specifications [156], a communications protocol providing a means for applying security to Web Services. All application data (i.e. privacy preferences, data collecting policies, context information, resource information, and privacy contracts) is managed using the native XML database Apache Xindice [157], which also runs under the Tomcat application server and provides a simple interface to saving and retrieving XML documents. The Xindice implements functionalities of the *Privacy Preferences Repository*, *Context Repository* and *Resource Repository* in the Privacy Agent. Jena [141] is a Java framework for building Semantic Web applications. It provides a programmatic environment for RDF [158], RDFS [132] and OWL [51], SPARQL [159] and includes a rule-based inference engine to support reasoning over OWL/RDF graphs. The Jena's built-in OWL reasoner and Generic Rule Engine are used to implement functionalities of the *Policy Evaluator* and *Preference Evaluator*, the two key components of the Privacy Agent, that perform semantic policy analysis of data collecting policies and privacy preferences, as well as perceiving context information.

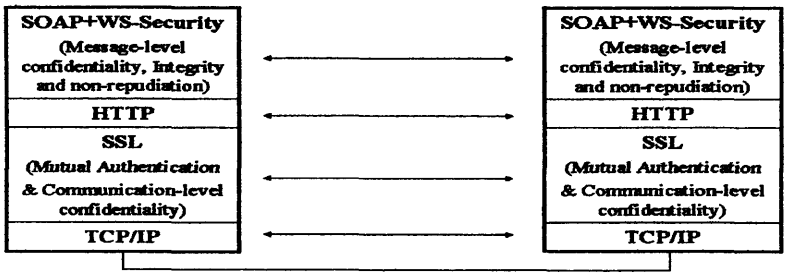

Figure 7.2 *A security protocol stack. Using secure communication and authentication via SSL running over TCP/IP, Privacy Agent, context-aware applications and users can exchange encrypted SOAP messages using WS-Security over HTTP*

```
POST /axis2/services/PrivacyAgentWS HTTP/1.1
User-Agent: Axis/2.0
SOAPAction: ReceivePolicy
Host: 127.0.0.1:8085
Transfer-Encoding: chunked
Content-Type: multipart/related;
boundary=MIMEBoundaryurn_uuid_9712F032FB365D971D11566009235521; type="application/xop+xml";
start="<0.urn:uuid:9712F032FB365D971D11566009235522@apache.org>"; start-info="text/xml";
charset=UTF-8


--MIMEBoundaryurn_uuid_9712F032FB365D971D11566009235521
content-type:application/xop+xml; charset=UTF-8; type="text/xml";
content-transfer-encoding:binary
content-id:<0.urn:uuid:9712F032FB365D971D11566009235522@apache.org>

<?xml version='1.0' encoding='UTF-8'?><soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header
/><soapenv:Body><x:Data xmlns:x="http://example.org/mtom/data"><xop:Include
href="cid:1.urn:uuid:9712F032FB365D971D11566009237523@apache.org"
xmlns:xop="http://www.w3.org/2004/08/xop/include"
/></x:Data></soapenv:Body></soapenv:Envelope>
--MIMEBoundaryurn_uuid_9712F032FB365D971D11566009235521
content-id:<1.urn:uuid:9712F032FB365D971D11566009237523@apache.org>
content-type:application/octet-stream
content-transfer-encoding:binary

<POLICIES xmlns="http://www.ee.ucl.ac.uk/~lezhang/PrivacyPolicy">
<POLICY name = "for Who-near-me application">
<ENTITY>
<APPLICATION> Who-near-me </APPLICATION>
<ID>0101 </ID>
<SERVICE-PROVIDER> O2 Mobile Operator </SERVICE-PROVIDER>
<CATEGORY> Tourism </CATEGORY>
<REQUST-MODE> continuous </ REQUST-MODE >
<SERVICE-MODE> optional </ SERVICE-MODE >
</ENTITY>
<STATEMENT>
<CONSEQUCNE> We try to help you find if any of your friends, colleagues, alumni are in the
city. We ask for your location information and social contact to make the service
available, and will retain them until you opt out of the service. </CONSEQUCNE>
<PURPOSE> navigation </PURPOSE>
<RECIPIENT> ours </RECIPIENT>
<RETENTION> stated-purpose </RETENTION>
<DATA-GROUP>
<DATA ref="#personalinfo.identity.realname">
<OPTION>
<TARGET-DATA ref="#personalinfo.socialgroup.friend">
<TARGET-DATA ref="#personalinfo.socialgroup.colleague">
<TARGET-DATA ref="#personalinfo.socialgroup.alumni">
</OPTION>
<OPTION>
<DATA ref="#personalinfo.location.city">
<DATA ref="#personalinfo.location.city.district">
<DATA ref="#personalinfo.location.city.district.street">
</OPTION>
</DATA-GROUP>
</STATEMENT>
</POLICY>
</POLICIES>


--MIMEBoundaryurn_uuid_9712F032FB365D971D11566009235521--
```

Figure 7.3 *A data collecting policies file (line 20-50) is sent as a MIME attachment of a SOAP message*

Secure communication among various parties in the distributed privacy protection model is achieved by implementing mutual authentication using the SSL protocol, in addition to encrypting and digital signing messages exchanged using OASIS WS-Security specification [156]. Figure 7.2 shows a secure protocol stack in the proof-of-concept implementation, while Figure 7.3 demonstrates a data collecting

policy file sent by the *Who-near-me* service to the Privacy Agent.

Employing ontology-based techniques is a key approach in the author's proposal to facilitate individual privacy expression and to automate privacy control process. The prototype implementation also demonstrates how various ontologies and ontology-based reasoning mechanisms developed in Chapter 5 are employed in the preference conflict and redundancy detection process, and in the policy evaluation process. Figure 7.4 below illustrates interaction models between human users and a Privacy Agent, and between a Privacy Agent and context-aware applications. Privacy Agent acts as a web service when context-aware applications request personal information, and when human users query, update and delete their privacy preferences that are stored in the *Privacy Preferences Repository* in the Privacy Agent.

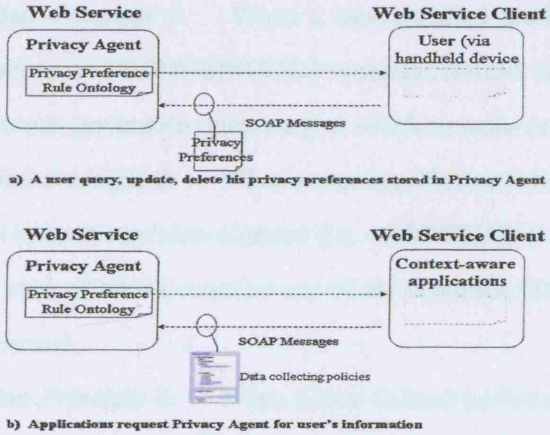

Figure 7.4 *Interaction models between a human user and its Privacy Agent and between Privacy Agent and context-aware applications*

As illustrated in Figure 7.4, the Privacy Preference Rule Ontology (in an OWL file) resides in the Privacy Agent and is made available to its users via personal handheld devices (such as PDA, smartphone, etc). The OWL file contains all classes and properties that are required to construct privacy preference rules, but does not include preference rule instances. The preference rule instances are dynamically generated when users specify manually their privacy preferences. Each time a user edits his privacy preferences, his personal handheld device refers to the Privacy Preference Rule

Ontology that is preloaded to the device's memory. The ontology specifies ranges of a set of values (e.g. Target, Location, Recipient, etc.) that the user can choose and the Privacy Agent can understand. Once a new privacy preference is created and sent (in the format of a SOAP message) to the Privacy Agent, the Privacy Agent invokes the privacy preference rule OWL file through Jena API, and automatically transforms the user-specified privacy preference into one or more atomic preference rule instances. The atomic preference rules instances are then stored in the *Privacy Preferences Repository* after the conflict detection and resolution process. The author has developed four principles to transform user-specified privacy preferences to atomic rules that comply with an ontology description of the Privacy Preference Rule Ontology. The transformation principles can be written using XSLT technology [169] and are presented as follows:

Transformation Principle 1: When a user-specified preference rule (defined as a <RULE> block in a <PREFERENCES>) contains multiple data elements, it is divided into several atomic preference rules, each of which contains one of the data elements.

Transformation Principle 2: When a user-specified preference rule contains the "or" operand in its context condition element (i.e. <CONTEXT>), it is divided into multiple atomic rules, each of which contains one of the condition elements that are connected by the "or" operand.

Transformation Principle 3: When a user-defined preference rule contains multiple values in its Purpose condition[25], it is divided into multiple atomic rules, each of which contains one of the values of the Purpose condition element.

Transformation Principle 4: The "!=" operand in a condition element (either policy condition or context condition) in the user-defined preference rules is expressed using notEqual() primitive in the corresponding atomic rules that are written using the Jena Generic Rule Expression.

In a similar way, data collecting policies from context-aware applications are sent to the

[25] As illustrated in the XML schema in Appendix E, only the Purpose condition element is allowed to have multiple values, but not the Retention and Recipient condition elements.

Privacy Agent in XML format and encoded in SOAP messages. Since the data collecting policies comprise vocabulary (e.g. Purpose, Recipient, Retention, etc) that is defined in the Privacy Preference Rule Ontology, the Privacy Agent can understand and handle them without ambiguity. Like user-defined privacy preferences, data collecting policies received from data context-aware applications need to be manipulated into a format that can be used in the policy evaluation process. The author has developed four transformation principles to transform data collecting policies received from the data requestor into policy rules that comply with the specification description of the Privacy Preference Rule Ontology. The transformation principles can be written using XSLT technology [169] and are presented as follows:

Transformation Principle 1: In a data collecting policy, only the elements enclosed in <STATEMENT> block are manipulated to form atomic policy rules and used for policy evaluation. These elements include <PURPOSE>, <RECIPIENT>, <RETENTION>, <DATA-GROUP>, <OPTION>, <DATA>, and <TARGET-DATA>.

Transformation Principle 2: The <PURPOSE>, <RECIPIENT>, <RETENTION> elements correspond to the policy condition element in the corresponding atomic policy rule (i.e. <PURPOSE>, <RECIPIENT>, <RETENTION> respectively). The <TARGET-DATA> element in the data collecting policy corresponds to the <TARGET> context condition element in the corresponding atomic policy rule, while <DATA> element corresponds to the <DATA> element in the corresponding atomic policy rule.

Transformation Principle 3: When a data collecting policy contains multiple values in its Purpose condition, it is manipulated to form multiple atomic policy rules, each of which contains one of the values of the Purpose condition element.

Transformation Principle 4: When a data collecting policy contains multiple <DATA> (or <TARGET-DATA>) elements) in a <OPTION> block, it is manipulated to form multiple atomic policy rules, each of which contains one of <DATA> (or <TARGET-DATA>) elements that are enclosed in the <OPTION> block.

7.2 Quantitative Performance Evaluation of Semantic Reasoning

Employing ontology-based techniques to represent and reason privacy preferences and contextual knowledge is a core part of the proposed Privacy Agent technology. It facilitates automated process of privacy control, and helps achieve the ultimate goal of the author's privacy protection solution —— to delivery relatively unobtrusive privacy management, and to empower people manage their privacy towards dynamic context-aware ubiquitous computing environment with relative ease.

In Chapter 6, the author described in detail how the ontology-based techniques work for semantic policy representation and reasoning in addition to context modeling and reasoning. In this section, a series of experiments are described to validate approaches taken by the author, as discussed in the introduction to this chapter. The evaluation work looks into the semantic reasoning in three processes respectively, namely, the privacy policy evaluation process, the preference conflict and redundancy detection process, and context reasoning process. The size and complexity of ontology, the complexity of inference rules, and the number of instances that will be reasoned over are three key factors measured in the performance evaluation.

7.2.1 Experimental Setup

During experiments, the Privacy Preference Rule Ontology is employed in privacy policy evaluation process and conflict and redundancy detecting process, while Context Ontology, that comprises Personal Information Ontology, Location Ontology, Time Ontology and Activity Ontology, are used for context reasoning. Table 7.1 summarizes TBox[26] information of these ontologies, including the number of classes, properties, and inherent individuals. The TBox information indicates the size and

[26] *TBox* contains the set of axioms that define concepts, properties and their various relationships in a knowledge base or KB. A counterpart is *ABox* which contains assertions about specific individuals [168].

complexity of the ontology. Note that unlike the Privacy Preference Rule Ontology, which contains a full definition of concepts (i.e. classes) and semantic relationships (i.e. properties) used to fulfill the task of the privacy policy and preference evaluation, the Context Ontology includes only necessary classes and properties to perform context reasoning, and for demonstration and evaluation purpose. The Context Ontology can be extended by incorporating it with other existing ontologies.

Table 7.1 *TBox Setting of the Privacy Preference Rule Ontology and Context Ontology*

	Privacy Preference Rule Ontology	Context Ontology			
		Personal Information Ontology	Location Ontology	Time Ontology	Activity Ontology
Classes	14	33	18	8	5
Properties	26	6	19	22	6
Predefined Instances (in TBOX)	25			14	
File size	35K Bytes	13K Bytes	16K Bytes	14K Bytes	6K Bytes

Experiments are carried out based on the prototype implementation of the *Policy Evaluator* and *Preference Evaluator* in the Privacy Agent. As described previously, both evaluators are built using Jena2 Semantic Web Toolkit [141], which provides a built-in OWL reasoner to support ontology-based description logic, and provisions a Generic Rule Reasoner to support rule-based inference over ontology description. The Jena OWL reasoner has three configurations (i.e. full version, mini version, and micro version), each of them is intended to be a sound implementation of a subset of OWL-Full[27] semantics, but none of them is complete [167]. In the experiments, OWL-Micro reasoner is used.

All experiments are conducted on an IBM laptop that has a hardware configuration of 512 MB RAM with PIII 1000MHz, and uses Windows 2000 operating system. Results are calculated as the average of twenty-five runs.

[27] See section 6.1.2 for more information about the OWL-Full sublanguage.

7.2.2 Experiments on the Policy Evaluation Process

In the policy evaluation process, the Privacy Agent compares data collecting policies received from data collectors with a person's privacy preference rules. A key step is to select semantically matched preference rules that are stored in the *Privacy Preferences Repository* and can be applied to evaluate data collecting policies. This process starts by loading the Privacy Preference Rule Ontology and the predefined inference rules into memory, and then uses the inference rules to reason over the ontology description of the preference rules that are stored in the *Privacy Preferences Repository* for semantic analysis. As presented in section 6.5, the author has developed a full set of 55 inference rules using Jena Generic Rule Expression to describe various relations specified in the Privacy Preference Rule Ontology, so as to deal with semantic reasoning to find fully-matching and partially-matching preference rules[28]. Among the 55 inference rules, 35 inference rules are used to conduct reasoning to find fully-matched applicable preference rules, while 47 inference rules are used to find partially-matched applicable preference rules. They share 27 inference rules that define equalToOrContain_p and equalToOrContainedBy_p relations.

Experiments show that the loading time of a full set of 55 predefined inference rules is 653 ms, while loading the Privacy Preference Rule Ontology (only TBox, i.e. without any preference rule instances in ABox[29]) causes 803ms. Table 7.2 and Figure 7.5 below summarize the file size and loading time of the Privacy Preference Rule Ontology with increasing size of ABox. The different set of TBox and ABox will be used in further experiments. The sum of loading time of the inference rules and the ontology accounts for the preparation time of the policy evaluation process. For example, the loading time of the Privacy Preference Policy Ontology with 50 preference rule instances is 935ms.

---

[28] Appendix F presents a full list of these inference rules.
[29] In a knowledge base, *ABox* contains assertions about specific individuals [168].

Table 7.2 *File size and loading time of the Privacy Preference Rule Ontology with the increasing size of ABox*

TBOX	The Privacy Preference Policy Ontology					
ABOX	0 preference rule instance	10 preference rule instances	20 preference rule instances	30 preference rule instances	40 preference rule instances	50 preference rule instances
Ontology file size	35K Bytes	48K Bytes	61K Bytes	72K Bytes	83K Bytes	93K Bytes
Loading Time	803ms	834ms	863ms	882ms	901ms	935ms

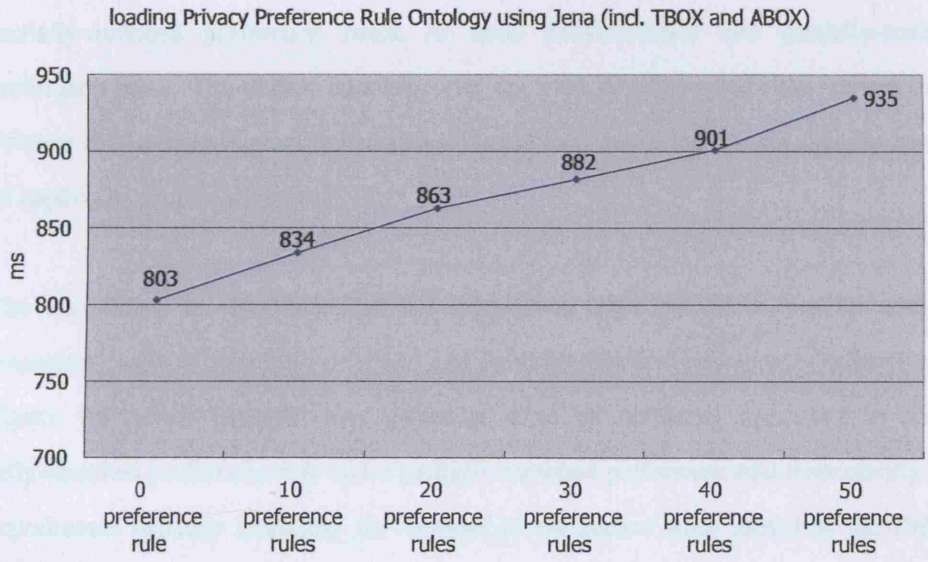

Figure 7.5 *The loading time of the Privacy Preference Rule Ontology with an increasing size of ABox. The horizontal axis is not meant to scale linearly to the number of preference rules, it just reflects six preference rule cases, and lines connecting two cases are only for illustration purpose.*

Recalling the Policy Evaluation Algorithm described in section 6.5.2, the data collecting policies received from data context-aware applications need to be manipulated into multiple policy rules before they can be evaluated against privacy preference rules in the policy evaluation process. As a result, the process time of the policy evaluation process depends on the complexity of data collecting policies received, in addition to the complexity involved in the process of selecting applicable preference rules to

evaluate policy rules transformed. The complexity of data collecting policy is measured by the number of policy rules they are transformed into.

In the experiments, the author takes the data collecting policy stated by the *Who-near-me* (as exemplified in Figure 5.1 in section 5.1.2) as a typical case, and assumes that each data collecting policy received by the Privacy Agent will be transformed into, on average, 12 policy rules. It is also assumed that, according to the policy evaluation algorithm (presented as Figure 6.18 in section 6.5.2), evaluating each policy rule would lead to either one fully-matched preference rule, or at most three partially-matched preference rules, or none fully-matched and partially-matched preference rules. The author carefully sets up a set of preference rules stored in the *Privacy Preferences Repository* so that different tests could lead to the specific number of applicable preference rules.

The experiment results show that the approaches taken by the author for semantic reasoning can find both fully-matched and partially-matched preferences rule correctly. Figure 7.6 below presents the execution time of semantic reasoning to find a fully-matched preference rule and a partially-matched preference rule respectively. The experiments increase gradually the number of preference rules stored in the *Privacy Preferences Repository* from a starting number of 10 to 50. More than 50 preference rules to be set by an individual are considered as a less likely case in real-world scenarios. The initial 10 preference rules are listed as follows. They are also used as an initial set in other experiments discussed in the following sections.

***Rule1**: My home contact information is allowed to disclose to my colleagues.*

***Rule2**: My contact information is forbidden to disclose to my colleagues.*

***Rule3**: My home contact information is forbidden to disclose to all my social group members.*

***Rule4**: My contact information is forbidden to disclose to all my social group members.*

***Rule5**: My location is allowed to disclose to my colleagues when I am not working.*

***Rule6**: My exact location is forbidden to disclose to all my social group members when I am in*

travelling.

Rule7: *My exact location is allowed to disclose to data collectors if the information retention is under applicable legal requirement.*

Rule8: *My location is forbidden to disclose to data collectors for the develop purpose.*

Rule9: *My location is allowed to disclose to data collectors for the develop purpose if the information retention is under applicable legal requirement.*

Rule10: *My exact location is forbidden to disclose to my colleagues for the navigation purpose when I leave London city.*

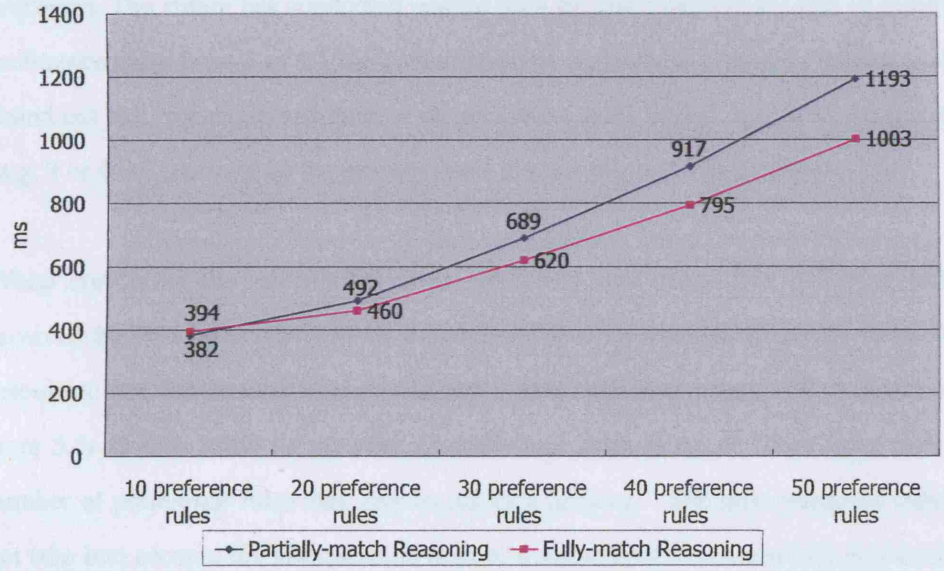

Figure 7.6 *Execution time of semantic reasoning to find a fully-matched preference rule and a partially-matched preference rule. The horizontal axis is not meant to scale linearly to the number of preference rules, it just reflects five preference rule cases, and lines connecting two cases are only for illustration purpose.*

As shown in the Figure 7.6, the increasing number of preference rules set by a person and stored in the *Privacy Preferences Repository* (i.e. the ABox size of the Privacy Preference Rule Ontology) has effect on the execution time of both fully-matched reasoning and partially-matched reasoning. When looking at the increase on the execution time, the effect of partially matched reasoning is greater than of fully matched reasoning process. This indeed is in correspondence to the fact that semantic

reasoning involved in partially-matching is more complex than that in fully-matching reasoning. As illustrated in Figure 6.16 and 6.17 in Chapter 6, selecting a partially-matched evaluation rule involves two more steps than finding a fully-matched preference rule. It needs to retrieve the Privacy Meta Policy set by users and comparing it with the preference rule's behavior.

Not presented in the Figure 7.6, the experiment also showed that the number of resulting preference rules obtained through partially-matched reasoning and fully-matched reasoning does not appear to effect the process time of both reasoning processes. The author has conducted several tests by increasing the number of resulting preference rules from 1 to 10, the performance of semantic reasoning is robust, as it is found out that, for any given number of preference rules loaded, only a trivial increase (e.g. 3 or 4ms) occurred on the process times that are presented in the Figure 7.6.

When combining the assumption made previously that every data collecting policy received by Privacy Agent will be transformed into on average 12 policy rules, it is calculated that the process time to evaluate a data collecting policy will at most range from 5.5s to 6.0s when the number of preference rules is set to 20 (a more realistic number of preference rules that may be set by a person). The time measured here do not take into account the manipulation time of transforming data collecting policies into multiple policy rules based on the specification of the Privacy Preference Rule Ontology. The result is acceptable for non-critical applications, such as *Who-near-me* service and *Personalized Shopping Guide* service exemplified in the use scenario in Chapter 4, especially when they run the first time. For more time critical applications (e.g. the *Emergency Notification* service exemplified in the use scenario) that demand immediate service response, the process time can be reduced if the Policy Evaluation Algorithm conducts the evaluation of policy rules in parallel, and/or a faster processor is used.

7.2.3 Experiments on the Conflict and Redundancy Detecting Process

In the author's proposal, the Privacy Agent detects potential conflict and redundancy arising from preference rules when users edit new privacy preferences, or update and delete existing preferences. The semantic reasoning involved in this process employs a set of predefined inference rules to reason over the ontology description of newly specified privacy preference rules to determine if they conflict or have redundancy when compared with the preference rules already stored in the *Privacy Preferences Repository.*

As presented in section 6.4, the author has developed a full set of 271 inference rules using Jena Generic Rule Expression to describe various relations specified in the Privacy Preference Rule Ontology, and to deal with semantic reasoning to detect potential conflicts and redundancy.[30] In reality, however, not all the inference rules are often exploited in the conflict and redundancy detection process. In order to measure the impact of the size of predefined inference rules on the performance of the conflict and redundancy detection process, two abbreviated versions, Micro (detectionrules_micro) and Mini (detectionrules_mini), are developed. The Micro version contains 190 inference rules, and is a subset of the full set of 271 inference rules. It includes inference rules that are able to perform semantic reasoning on preference rules that have at most two subsumption relations in their context conditions. The Mini version contains 81 inference rules, and is a subset of the Micro version. It includes inference rules that are only able to perform semantic reasoning on preference rules that at most have two context conditions. Table 7.3 below summarizes the number of inference rules that are used to define each version.

Experiment results show that the loading time of the Full set, Micro set and Mini set of

[30] The author presents in Appendix E the most common inference rules that are used to detect preference conflict and redundancy.

the predefined inferences rules cause 736ms, 711ms, and 675ms respectively. The sum of loading time of the inference rules and the Privacy Preference Rule Ontology (as obtained through experiments in previous section 7.2.2) accounts for the preparation time of the conflict and redundancy detection process.

Table 7.3 *The number of inference rules that are used to define the Full set, Micro set, and Mini set*

	Full set	Micro set	Mini set
inference rules to define Modality Conflicts	1	1	1
inference rules to define NonModality Conflicts	11	11	11
inference rules to define Redundancy	6	6	6
inference rules to define equalTo_p relations	8	8	8
inference rules to define equalTo_c relations	16	16	11
inference rules to define subsume_c & subsumedBy_c relations	133	124	32
inference rules to define mutuallySubsume_c & mutuallySubsumedBy_c relations	96	24	12
total number of inference rules	271	190	81

To evaluate the performance of semantic reasoning in the conflict and redundancy detection process, three further experiments were carried out separately to measure the impact of three key factors: the complexity of inference rules, the size of privacy preference rules that are stored in the *Privacy Preferences Repository* and meant to be reasoned over, and the complexity of the semantic reasoning process. The three experiments simulate the case that a person edits a new preference rule and the Privacy Agent tries to detect potential conflicts or redundancy when updating it into the *Privacy Preferences Repository.* In first and second experiments, the new preference rule is set to have one data element, one policy condition (of Purpose, Retention, Recipient elements), and two context conditions (of Target, Location, Time, Activity elements). A

preference rule with such a setting represents a case with average complexity[31], and can be reasoned by using even the smallest set of inference rules (i.e. Mini version). In the third experiment, however, the new preference rule is set with various combinations of context and policy conditions, which represent varying levels of complexity of preference rules.

The first experiment evaluated the impact of the complexity of inference rules. Figure 7.7 below presents the execution time of semantic reasoning to find a Non-Modality conflicting rule using Micro and Mini version of inference rules. The experiment chose the non-modality conflict reasoning to evaluate as it is more complex than semantic reasoning to detect modality conflict and redundancy. The author also carefully set up a set of preference rules stored in the *Privacy Preferences Repository*, to ensure that they can be reasoned over by both Micro and Mini set of inference rules.

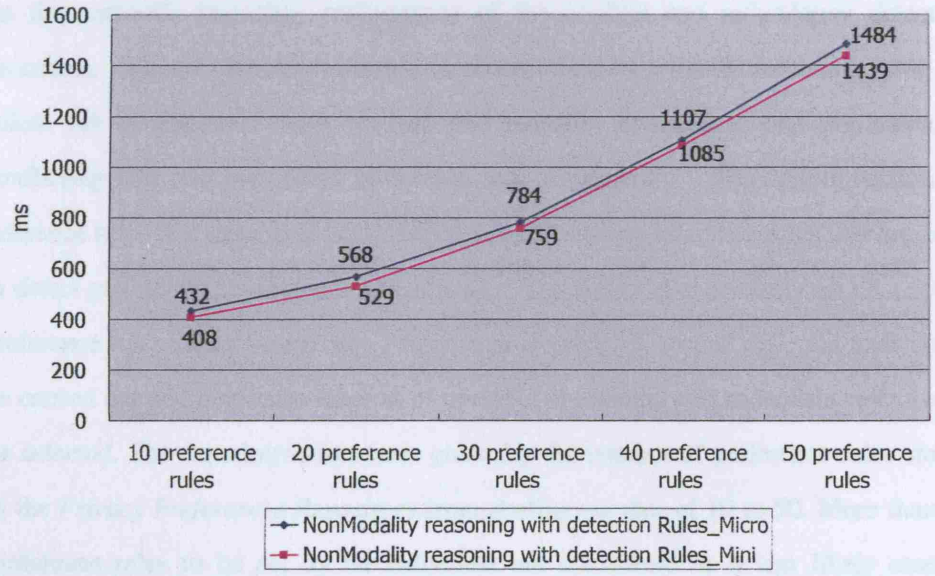

Figure 7.7 *The execution time of semantic reasoning to find a Non-Modality conflicting rule using Micro and Mini version of the inference rules. The horizontal axis is not meant to scale linearly to the number of preference rules, it just reflects five preference rule cases, and lines connecting two cases are only for illustration purpose.*

[31] A preference rule with average complexity could be, for example, that my location data (Data) should not be disclosed to my colleagues (Target) for the purpose policy condition (Purpose) declared in the Who-near-me service once I leave London city (Location).

It is shown in Figure 7.7 that the difference between using Micro and Mini set of inference rules in all cases is small and is independent of the number of preference rules, therefore the number of inference rules do not appear to have obvious impact on the process time of semantic reasoning to detect Non-Modality conflict. An add-on test was carried out to measure the effect of the complexity of inference rules in addition to their size. The author increased the mini set of inference rules to the size of Micro set (i.e.190 rules) by filling it with repeated inference rules. It was discovered that the process time is unchanged compared with the original mini set (i.e. 81 rules). As a result, it is reasonable to conclude that it is the complexity of inference rules but not the number of inference rules that have impact on the process time, although the impact is small.

The second experiment evaluated the impact of the number of privacy preference rules on the semantic reasoning performance of the conflict and redundancy detection processes. Figure 7.8 below presents the process time for semantic reasoning using the Micro set of inference rules to find one modality conflicting, one non-modality conflicting and one redundant preference rule respectively. The Micro version of inference rules is chosen as it comprises the most common inference rules that are used to detect preference conflict and redundancy. The author also carefully set up a set of preference rules stored in *Privacy Preferences Repository*, so that different tests could be carried out and a specific amount of potential conflicting and redundant rules could be detected. The experiment increases gradually the number of preference rules stored in the *Privacy Preferences Repository* from starting number of 10 to 50. More than 50 preference rules to be set by an individual are considered as a less likely case in real-world scenarios.

The experiment results validate the approach taken by the author since semantic reasoning can detect non-modality conflicting, modality conflicting and redundant preference rules correctly. Similar to the performance evaluation results of the policy

evaluation process, Figure 7.8 shows that the increasing number of preference rules set by users and stored in the *Privacy Preferences Repository* (i.e. the ABox size of the Privacy Preference Rule Ontology) has effect on the execution time of non-modality reasoning, modality reasoning and redundancy reasoning. When looking at the increase on the execution time, the effect of non-modality conflict reasoning is greater than that of redundancy reasoning, which is in turn greater than that of modality conflict reasoning. This indeed is in correspondence with the fact that the semantic reasoning involved in partially-matching is more complex than that in fully-matching reasoning. As illustrated in Figure 6.12 and 6.13 in Chapter 6, detecting a non-modality conflict involves at least one subsumption reasoning while performing the modality conflict detection may need none. Also, the non-modality conflict detection process involves more inference rules than that required for the redundancy detection.

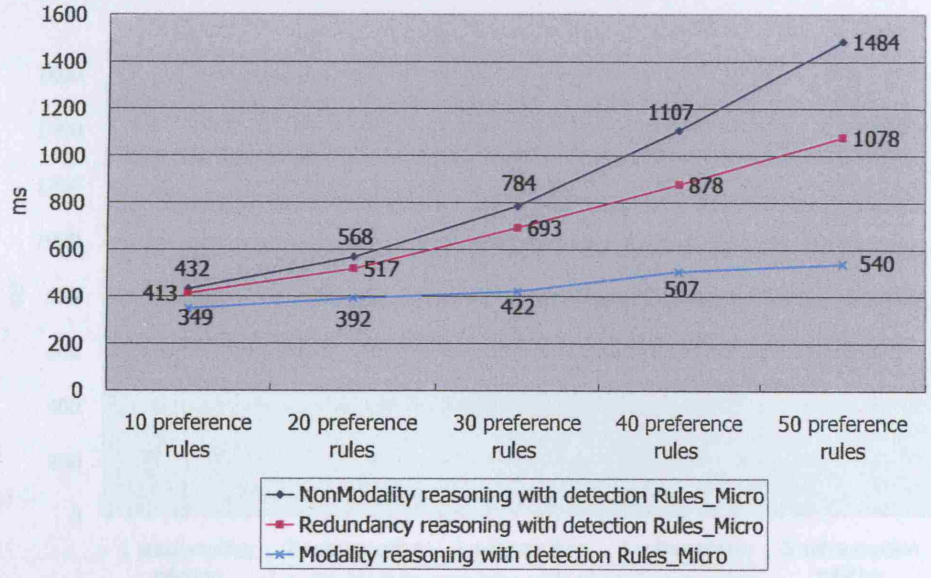

Figure 7.8 *Execution time of semantic reasoning to find respectively one non-modality conflict, one modality conflict, and one redundancy using the Micro version of inference rules. The horizontal axis is not meant to scale linearly to the number of preference rules, it just reflects five preference rule cases, and lines connecting two cases are only for illustration purpose.*

Not presented in the Figure 7.8, the second experiment also found that the number of resulting preference rules obtained does not appear to effect the process time of the

conflict and redundancy detection process. The author has conducted several tests by increasing the number of resulting preference rules form 1 to 10, the performance of semantic reasoning is robust, as it is found out that, for any given number of preference rules loaded, only trivial increase (e.g. 3 or 4 ms) occurred on the process times that are presented in the Figure 7.8.

The third experiment was carried out to test if the performance of the preference conflict and redundancy detection process is subject to the complexity of semantic reasoning occurred. Varying levels of complexity are tuned by carefully setting up data and condition elements of preference rules so that a specific amount of subsumption reasoning is required when evaluating them. In the preference rule evaluation process, a maximum of 5 subsumption reasoning may occur when two preference rules have their disclosed data, target, location, time, activity conditions all subsumed by each other.

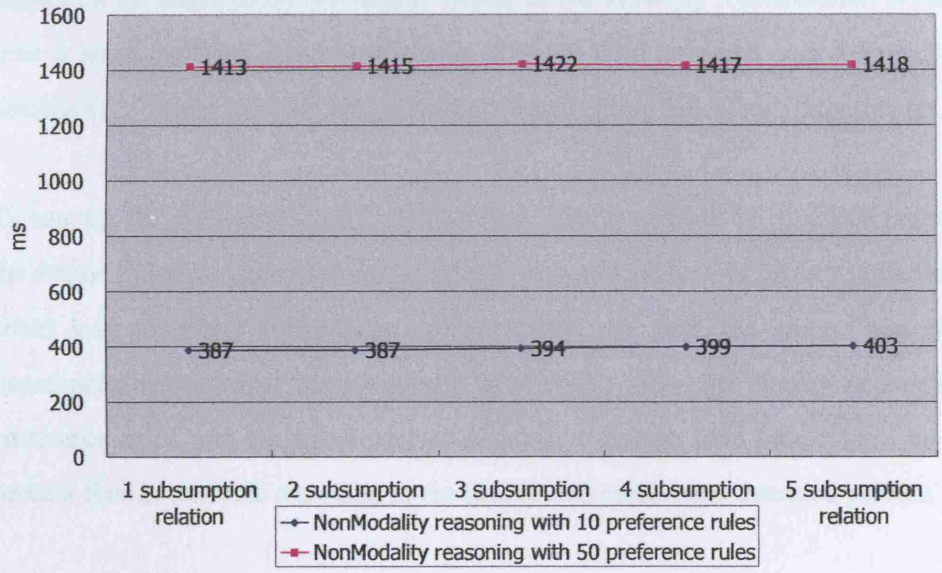

Figure 7.9 *Process time of semantic reasoning to find a non-Modality conflicting rule when the size of the Privacy Preferences Repository is set to 10 and 50 respectively. The horizontal axis is not meant to scale linearly to the number of subsumption relation, it just reflects five subsumption relation cases, and lines connecting two cases are only for illustration purpose.*

Figure 7.9 above presents the process time of semantic reasoning to find a Non-Modality conflicting rule when the size of preference rules stored in the *Privacy Preferences Repository* is set to 10 and 50 respectively. This experiment is conducted using Micro version of inference rules.

It is shown in Figure 7.9 that the semantic reasoning over 5 subsumption relations does not necessarily cause more time than the reasoning over 1 subsumption relation, which indicates the robust performance of semantic reasoning. The third experiment also confirms the observation from the second experiment that the size of preference rules has an important effect on the process time of semantic reasoning. Note that the third experiment looks into only the most important factor involved in evaluating the complexity of semantic reasoning in the policy evaluation process. Other factors may include the depth of subclass hierarchy in the Privacy Preference Rule Ontology, and the process time for equation reasoning. These factors are considered to have less impact on the performance evaluation results, as the hierarchy representation is rarely over 5 levels, and the equation reasoning does not need to reason over the ontology description. These factors have not been measured during the author's experiments.

To sum up, the evaluation work conducted here helps to validate the approach taken by the author and gives some confidence that the semantic analysis of privacy preferences could lead to robust performance. As shown in the first, the second and third experiments respectively, the complexity of inference rules, the number of resulting preference rules, and the complexity of semantic reasoning have little impact on the process time of semantic reasoning in the conflict and redundancy detection process.

A great impact on performance is made by the number of preference rules that are set by individuals and meant to be reasoned over. Larger numbers of preference rules not only causes more preparation time, but also increases prominently the process time of semantic reasoning in the detection process. In addition, the second experiment shows that the impact of the size of preference rules varies with different types of reasoning

and is associated with the complexity of inference rules involved. These results correspond to the experimental results of the policy evaluation process, in which the number of preference rules also has an obvious and varying impact on its process time.

The experiments indicate that overall the semantic reasoning to detect preference conflict and redundancy costs less than 1.5 second in various cases. The result is acceptable, as more than 50 preference rules to be set by an individual is considered as a less likely case in real world scenarios. Also in reality, the preference rules may not be as complex as those used in the experiments, and only two or three subsumption reasoning are typically expected when evaluating a privacy preference rule.

7.2.4 Experiments on the Context Reasoning Process

As described in the working logic section in Chapter 4, the Privacy Agent needs to perceive its owner's context, in order to make appropriate information disclosure decisions to him/her. Experiments carried out in this section simulate the case in which hybrid semantic reasoning mechanisms that combine ontology-based Description Logic and rule-based application-specific logic are used to reason over four contextual ontologies (i.e. Personal Information Ontology, Location Ontology, Time Ontology, Activity Ontology) to derive a specific activity that is currently performed by a person. The two activity inference rules exemplified in the activity reasoning part in section 6.6 are used to conduct the experiments. For convenience, the author repeats the two inference rules as follows:

Inference Rule 1: *If Alice's current location is inside her working place and current time is within working hours, it can be inferred that she is working.*

Inference Rule 2: *If Alice's current location is not in the area of her hometown, and the schedule in her diary shows she may be in traveling today, it can be inferred that she is traveling.*

```
[InferenceRule1:
 (?p per:isEngagedIn act:working) ←
   (?p rdf:type per:person)
   (?p per:hasContextualLocation ?lp)
   (?lp rdf:type loc:ContextualLocation)
   (?p per:hasWorkingPlace ?com)
    (?com rdf:type loc:SpatialEntity)
   (?com loc:hasLocation ?lc)
   (?lc rdf:type loc:PremiseLocation)
   (?lp loc:locatedIn ?lc) ←
        (?lp loc:hasGeograhicAddress ?addressA)
        (?lc loc:hasGeograhicAddress ?addressB)
        (?addressA rdf:type loc:GeographicalAddress)
        (?addressB rdf:type loc:GeographicalAddress)
        (?addressA loc:relatedTo ?addressB)
   ]
```

```
[InferenceRule2:
  (?p per:isEngagedIn act:traveling) ←
   (?p rdf:type per:person)
   (?p per:hasContextualLocation ?lp)
   (?lp rdf:type loc:ContextualLocation)
   (?p per:hasHome ?home)
   (?home rdf:type loc:SpatialEntity)
   (?home loc:hasLocation ?lh)
   (?lh rdf:type loc:PremiseLocation)
   (?lp loc:disconnectedWith ?lh) ←
        (?lp loc:hasGeograhicAddress ?addressA)
        (?lp loc:hasGeograhicAddress ?addressB)
        (?addressA rdf:type loc:GeographicalAddress)
        (?addressB rdf:type loc:GeographicalAddress)
        noValue (?addressA loc:relatedTo ?addressB)
   (?p per:hasScheduledActivity ?act)
   equal(?act act:traveling)
   (?act act:HappenWhen ?dates)
   (?dates tme:intContain tme:today)
   ]
```

Figure 7.10 *Example rules used to configure HP Generic Rule Reasoner to support activity reasoning. (i.e. Figure 6.22)*

As presented in the section 6.6, perceiving a person's activity involves the combination of spatial reasoning, temporal reasoning, and retrieval of personal information. The author has developed 14 inference rules using Jena Generic Rule Expression for spatial reasoning, and 12 for temporal reasoning.[32] The sum of loading time for these inference rules and the four contextual ontologies accounts for the preparation time of the context reasoning process. Experiments show that the preparation time for the context reasoning is 1425ms, of which 838ms is for loading the context ontology (only TBox, i.e. without any instances in ABox), and 587ms is used to load all 28 inference rules.

Scalability has been identified by previous work [48] as a main issue when executing reasoning with ontological languages to conduct context reasoning. Two experiments were carried out separately to address that concern in this work. The first experiment aimed to evaluate the performance of context reasoning with a growing number of instances added to the ABox of the Context Ontology. According to the inference rules illustrated in Figure 7.10, perceiving a person's activity involves various classes, including ActivityEntity, SpatialEntity, PremiseLocation, ContexutalLocation,

[32] Appendix G presents a full list of inference rules to conduce context reasoning in this work.

GeographicAddress, ScheduledActivity and Date. In this experiment, the number of instances of each of these classes was increased from 10 to 50.

Experimental results are shown in Figure 7.11. Execution time grows almost linearly with the number of instances added to the ABox. The semantic reasoning to perform inference rule 2 needs more time than was the case for inference rule 1. This is in accord with the fact that the inference rule 2 is more complex than the inference rule 1. As illustrated in the Figure 7.10, the inference rule 2 involves retrieving personal information and conducting spatial and temporal reasoning, while the inference rule 1 does not need to conduct temporal reasoning.

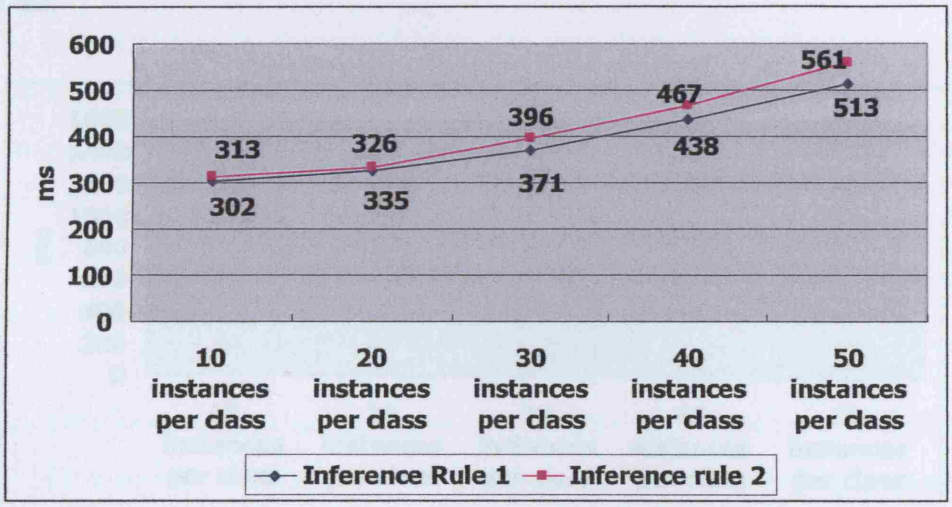

Figure 7.11 *Execution time of semantic reasoning to perform inference rule 1 and inference 2, with increasing number of instances added to the ABox of the Context Ontology. The horizontal axis is not meant to scale linearly to the number of instance per class, it just reflects five instances-per-class cases, and lines connecting two cases are only for illustration purpose.*

The second experiment is to measure the performance of the context reasoning with a realistic ontology. As explained in section 6.7, to avoid unnecessary complexity (that is more than required by most simple applications, and more than affordable by the hosts of the Privacy Agent that have limited CPU power, memory and computing facility), all individual contextual ontologies developed in this work only include concepts (i.e.

classes) and information relationships (i.e. properties) that are necessary for performing context perception and for demonstration purpose. In reality, ontologies may contain various domain knowledge and thus concepts and relationships that are not particularly relevant to a reasoning task. In this experiment the author simulates the real-world scenario by increasing both the TBox and ABox size of the Context Onotlogy. More specifically, the TBox is expanded to include 25 classes that are not involved in context reasoning, while the ABox is populated with about 800 instances (10 per class) belonging to classes that are not involved in the reasoning, and a growing number of instances that belong to classes involved in the reasoning. The classes involved in the reasoning is the same as that in the first experiment, i.e. ActivityEntity, SpatialEntity, PremiseLocation, ContexutalLocation, GeographicAddress, ScheduledActivity and Date.

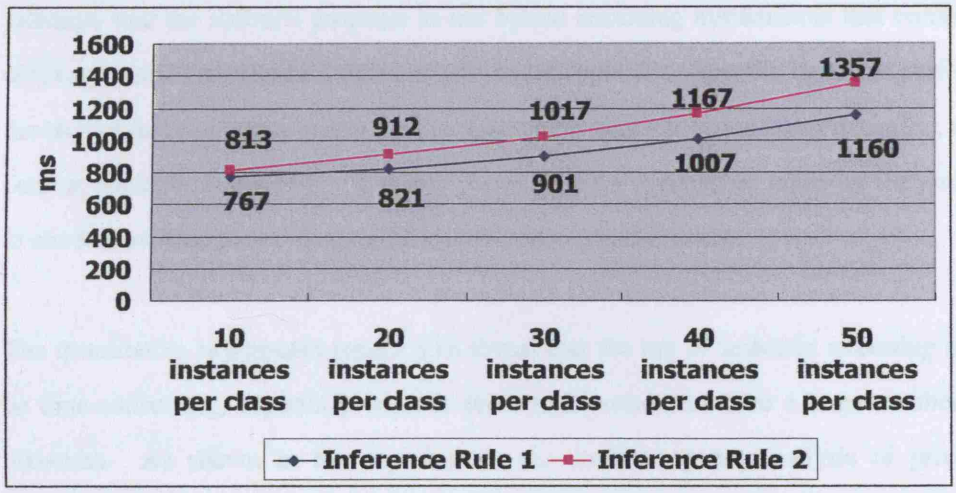

Figure 7.12 *Execution time of semantic reasoning to perform inference rule 1 and inference 2, with increasing number of instances added to both the TBox and the ABox of the Context Ontology. The horizontal axis is not meant to scale linearly to the number of instances per class, it just reflects five instances-per-class cases, and lines connecting two cases are only for illustration purpose.*

Figure 7.12 presents the result of the second experiment, in which the number of instances that belong to classes involved in the context reasoning grows from 10 to 50 gradually. Similar to the result of the first experiment, the execution time grows

linearly with the number of instances that belong to classes involved in the reasoning task. Also when looking at the increase in the execution time, the effect of enlarging the size of ABox on the process time of semantic reasoning to perform the inference rule 2 is greater than that is the case for the inference rule 1. When the TBox of the Context Ontology contains a large number of classes (around 100 in the author's setup) and the ABox is filled with a rather large number of instances (more than 1000), the execution time of context reasoning is of the order of a second, even if few instances belong to classes involved in the reasoning task.

7.2.5 Discussion of Quantitative Performance Evaluation

As a key contribution of the performance evaluation work, the experimental evidence validates that the author's proposal to use hybrid reasoning mechanisms that combine ontology-based Description Logic and rule-based application-specific logic can perform the task of privacy policy evaluation, preference conflict and redundancy detection, and context perception properly. It is also shown that the approaches taken by the author to conduct semantic reasoning could lead to a robust performance.

The quantitative experiment results also reveal that the use of semantic reasoning may be time-consuming, especially when it needs to be conducted over a large number of instances. As shown in the experiments, the semantic policy analysis of privacy preferences requires about 1.4 second when the size of preference rule instances is set to 50, while the context reasoning over two simple inference rules cost over 1 second when the size of ontology is big enough. Although the author argues that the size and complexity of preference rules set by an individual in real world scenarios may not be comparable to those employed in the experiments, the experimental result may have an important implication for critical applications (such as the *Emergency Notification* service exemplified in the use scenario in Chapter 4) that demand immediate response and especially when they play with resource-constrained devices. It is speculated that

the memory is a less stringent resource than processing power on reasoning performance. As illustrated in Table 7.2, the file size of the Privacy Preference Rule Ontology even with 50 preference rule instances is small (i.e. 93K Bytes).

However, the author's experience with using ontology and logic inference tools during the evaluation experiments has revealed that the potential of employing semantic reasoning to conduct semantic policy analysis of privacy preferences and context perception has been limited by both immaturity of ontology standards and a lack of tool support for reasoning axiomatic rules over ontology descriptions. In particular, OWL currently does not provide direct support for axiomatic rules, which limits the types of reasoning that are possible with OWL alone. The HP's Jena Semantic Web Framework [141] is chosen for the proof-of-concept implementation of the *Preference Evaluator* and *Policy Evaluator* and used in the evaluation work, as it provides an approach to overcome the above limitation of using the OWL language.

During the experiments, the author has attempted to explore alternative evaluation setups and tools to validate the consistency of results obtained using the Jena. This included exploring the use of JESS [160], a rule-based logic engine, to replace Jena Generic Rule Inference Engine, and employing Pellet OWL reasoner [161], apparently the most robust OWL-DL reasoner so far, to replace the Jena built-in OWL reasoner. But none of the attempts were successful. On the one hand, JESS does not provide programmatic support to work together with Ontology-based Description Logic that is a core part of the author's proposal to conduct semantic analysis. On the other hand, when using the Pellet-OWL reasoner with Jena Generic Rule Engine, experimental results exhibited strange behaviors, which were confirmed when executing experiments on a different machine. The odd results are probably due to incompatibility between the use of the Jena Generic Rule Engine and the Pellet-OWL reasoner. Another alternative investigated here was to use Pellet OWL reasoner with the SWRL (Semantic Web Rule Language) [162], an emerging standard developed by the DAML group to combine OWL with the Rule Markup Language (RuleML) [52]. But unfortunately the

Pellet's support for the SWRL is still under development. The combination is not able to conduct reasoning required for semantic policy analysis of privacy preferences and context perception.

The author thus suggests that future work should include further evaluation of improving ontology standards and tools, and exploration of alternative evaluation setups and tools as they become available, to extend the approaches developed here to perform semantic reasoning.

Scalability has been measured as a major issue when executing context reasoning. The experiments conducted in section 7.2.4 have indicated that the scalability issue can be addressed by encompassing in an ontology concepts (i.e. classes) and relationships (i.e. properties) relevant to a specific reasoning task, and by limiting the number of class instances. A major limitation of the author's context reasoning work comes from the simple inference rules used in the experiments. In this work, the author only exemplified simple inference rules based on common sense to show how the rule-based application-specific logic over ontology description works. In reality, there exists various ways to perceive human activities, and inference rules can be very complicated. The rule-based model might not scale well when the number of conditional factors increases in logical inference rules in order to achieve certain inference accuracy. Future work on evaluating the performance of context reasoning may include investigating the feasibility of incorporating more sophisticated logic and Artificial Intelligence techniques [164] in the context perception process, in particular, to define and apply inference rules. The future work must bear in mind the resource-constrained environments that the Privacy Agent will play with.

Aside from the work validating the semantic reasoning approaches, there is a need to evaluate other system and operational issues associated with applying the distributed protection model and the Privacy Agent technology in practical and real deployment scenarios. These issues include, but are not limited to, investigating the applicability of

different security mechanisms and enhanced trust models to improve the effectiveness of the distributed protection model, and conducting a formal usability study to understand how people perceive and use the Privacy Agent technology in realistic situations. These issues will be further discussed in section 8.2 Future Work.

7.3 Chapter Summery

This chapter demonstrated through prototype implementation how the author's privacy solution may work in real world situations. The implementation does not cover all aspects of the proposed solution, but focuses on privacy interactions between human users and a Privacy Agent, and between a Privacy Agent and context-aware applications, in addition to staple functionalities of the *Policy Evaluator* and *Preference Evaluator* in the Privacy Agent. The reasoning behind the technology choices and implementation considerations was discussed.

The chapter also presented the quantitative performance evaluation of the approaches taken by the author to conduct semantic reasoning over privacy policy and preference as well as context knowledge. The experimental evidence demonstrates the feasibility of performing semantic policy analysis of privacy preferences to preserve individual privacy towards dynamic context-aware computing environment, and provides insights into using ontology-based methods in real world scenarios that might have limited processing power and memory.

Chapter 8: Conclusions and Future Work

In this final chapter, the author summarizes novel aspects and major contributions of her doctoral work, and discusses some future research directions.

8.1 Novelty Aspects and Major Contributions

Privacy research in the field of context-awareness is still in its early stage and ongoing. An extensive literature review conducted by the author in Chapter 3 has indicated limitations in approaches taken to date and in particular it seems that personal privacy needs in dynamics context-aware environments remain largely unsatisfied. These limitations include:

- Most of the existing solutions are not user-centric but system-centric, in other words, they do not support active participation of individuals in controlling their own information, but are concerned mainly with access control and security mechanisms regarding the information that is kept within their systems.
- Existing solutions do not address the challenge of the distributed nature of context information, and simply assume personal information is controlled centrally by its owner or system administrator.
- Only a small subset of the privacy needs (as identified in section 3.5) had been adequately addressed, comprehensive privacy solutions did not exist.
- Limited efforts of privacy expression languages and mechanisms to facilitate individual privacy expression.
- Very simple user interfaces for privacy management in the form of information layout, not respond to dynamic context changes.

To address the above limitations, the author first introduced in Chapter 4 ***a new understanding of ubiquitous computing systems as a composite environment*** that

takes into account heterogeneous types of ubiquitous systems. The notion of composite ubiquitous computing environments presents a different view from existing solutions that either focus on constrained SmartSpaces, or only concern ubiquitous architecture based on specific network systems, such as the mobile network.

The author then discussed in section 4.2 that there are primarily five parties (i.e. Infrastructure Provider, Context Provider, Service Provider, Privacy Agent and Human User) involved in preserving privacy in the composite ubiquitous computing environment, and argued that the fact that multiple roles are played by single organization has important implications for preserving privacy in such an environment, as a clear separation of accountability of different parties involved in privacy protection may be hard to establish. To tackle this problem, the author introduced a ***distributed privacy protection model*** that separates the privacy decision process from the enforcement process and allows multiple parties to participate in information disclosure control with a clear accountability. The distributed privacy protection model takes into account the dispersed nature of context information in dynamic context-aware ubiquitous computing environments, and overcomes the limitations of exiting solutions that are based on many simplifying assumptions, e.g., they only consider location information, or assume that context information is neatly partitioned into repositories that are under the control of a single user.

Central to the distributed privacy protection model is the author's work on the ***intelligent agent technology*** that handles privacy-related interactions on behalf of individuals. It aims to facilitate a relatively unobtrusive user participation in controlling the disclosure of their sensitive information, and addresses two key concerns to preserve privacy in context-aware ubiquitous computing environments: privacy feedback (i.e. notifying people of relevant information disclosure) and privacy management (i.e. allowing people to express their privacy preferences and manage privacy levels). The Privacy Agent technology also provides mechanisms to support ambiguous disclosure and some levels of plausible deniability, two key privacy requirements that have not

been adequately supported by many existing solutions.

An important approach has been taken by the author to develop the ***Privacy Agent technology*** by grounding its design guidelines on the principles of the Fair Information Practices (in particular, *Notice*, *Choice and Consent*, *Data Access and Management*, and *Data Security*). The author recognized the difficulty of providing a perfect privacy protection (if there is such a thing) by employing technical mechanisms alone, and strived to incorporate legal and social requirements into the proposed technical solution. Integration of social and legal privacy considerations as proposed in this work do not appear to have been emphasized to date. In addition, unlike many existing solutions, the development of the Privacy Agent technology is independent of specific context-aware architectures, systems and middleware solutions. The proof-of-concept implementation using Web Service technologies in Chapter 7 demonstrated that the solution is easy to deploy to achieve interoperability across different system platforms and devices, and can be scalable to the global Internet.

To facilitate automated processes of the Privacy Agent, the author has proposed a new lightweight ***Privacy Policy/Preference Language***, so that various parties involved in the distributed privacy protection model and functional components of the Privacy Agent can have a common understanding about privacy requirements while interacting with each other. The development of the syntax and semantics of the language has adapted the terminology and policies specified in W3C's Platform for Privacy Preferences Project (P3P) specification, in order to benefit from the substantial legal and social expertise that has been put into the development of the P3P standards. However, unlike the P3P policy and previous researches that apply the P3P in ubiquitous computing environments, the Privacy Policy/Preference Language developed by the author does not limit its practice only to be a vehicle for data collectors to state their collecting requirements. It instead provides an integrated solution for specifying data collecting policies and individual privacy requirements. More importantly, the language can be used to construct specific preferences to limit information disclosure not only

with respect to the data collecting policies, but also in response to dynamic contexts.

As another key novelty aspect of the privacy solution proposed, the author has exploited ***ontology-based methods*** to model the Privacy Policy/Preference Language developed in Chapter 5, and explored novel hybrid reasoning mechanisms that combine ontology-based Description Logic and rule-based application-specific logic for semantic analysis of individual privacy preferences and context reasoning. To the best of the author's knowledge, the semantic reasoning approaches taken by the author to perform privacy policy evaluation as well as to detect preference conflict and redundancy have not been presented in previous work. As a key contribution of this doctoral research, the author's experience with ontology-based methods has revealed the potential of taking advantage of semantically-rich policy representation and reasoning to reduce human error, simplify policy analysis, detect policy conflicts, and facilitate policy understanding.

The author demonstrated the feasibility of coupling rule-based method with the semantic web programming framework (i.e. Jena) to conduct quantitative performance evaluation on semantic reasoning approaches. The experimental evidence shown that the author's proposal of using hybrid reasoning mechanisms can perform the task of semantic privacy policy evaluation, preference conflict and redundancy detection, and context perception properly, and lead to robust performance. To the best of the author's knowledge, the ***evaluation work*** is among very few successful attempts that find methods to validate the ontology-based approaches, due to the difficulty caused by both immaturity of ontology standards and a lack of tool support for reasoning axiomatic rules over ontology description.

8.2 Future Work

The author presented through Chapter 4 to Chapter 6 novel approaches and key technologies that comprise the privacy solution proposed in this work, and demonstrated through the proof-of-concept implementation and quantitative performance evaluation in Chapter 7 how the proposed solution may work in real world scenarios and with appropriate performance. However, issues remain and should be addressed before the proposed solution is about to implement in practical and real deployment scenarios. The author suggests the following direction to take in future work:

- Enhancing security mechanisms and introducing trust model in the privacy protection model
- Conducting formal usability studies on the proposed privacy agent technology, including developing an actual user interface
- Improving the context reasoning capability by applying more sophisticated logic and advanced Artificial Intelligence techniques with the rule-based method.
- Continuing performance evaluation work with emerging standards and tools, and looking into other factors and operational issues.

- ***Enhancing security mechanisms and introducing trust model***

Security mechanisms are a staple element to secure the effectiveness of the distributed protection model and privacy protection solution proposed in this work. The author identified in Chapter 4 some potential security threats and trust compromise scenarios when distributing information disclosure decision-making and enforcement processes in the network, and discussed some possible safeguard and mitigation measures that should be considered when implementing the distributed model and intelligent agent.

The prototype implementation in Chapter 7 demonstrated the use of some encryption and cryptographic tools to secure communication in the distributed privacy protection

model, and to protect information confidentiality and integrity. However, it must be noted that the security mechanisms employed are necessary but not sufficient to secure the privacy protection. For one thing, the privacy protection model based on encryption and cryptographic tools is only as secure as security concepts held behinds the tools, but unanticipated mathematical developments may make them vulnerable to attack eventually. For another, as discussed in Chapter 4, there are some threat models (such as Single Point of Attack and Denial of Service attack) and trust compromise scenarios that are not unique to the distributed model proposed by the author, and their safeguard mechanisms are not addressed in this work. Instead, the author relies on existing solutions and leaves to implementers to select appropriate mechanisms to mitigate problems in certain system environment.

Future work to enhance security and trust mechanisms of the privacy protection model may include two aspects. The first aspect is to investigate the approaches to integrate other privacy mechanisms such as identity management tools through anonymity and pseudonymity into the privacy protection model. As presented in literature work in Chapter 3, previous work [66, 67] provided methods that employ pseudonym to hide a user's real identity and combine encryption techniques (such as digital signatures and blind signature) to help achieve anonymous yet authentic communication. Other more sophisticated pseudonymity solutions include [62, 69]. They proposed stronger pseudonym protection by building underlying anonymous communication infrastructure, so that anonymity can be achieved at the communication level.

The second aspect involves introducing trust acquisition and evaluation mechanisms into the privacy protection model. As discussed in section 4.5.3, two type of trust are involved in the distributed privacy protection model. The first type is human trust, in particular, how much people trust their privacy agent to make the "right" decision that conforms to their expectation and wills. Literature work in section 3.3.5 presented research efforts that applied computational trust mechanisms to enhance human trust in various system and application environments, which provided insight to enhance human

reliance on their Privacy Agent. However, the problem of validating the computational trust mechanisms that attempt to mimic human notion of trust is hard (if not impossible) to resolve. The second type of trust is machine trust, in particular, how the Privacy Agent selects and evaluates if a service provider, context provider, or privacy policy enforcement point (PPEP) is trustful. The methods to acquire machine trust have been proposed by some related work. For instance, Seng and Arbaugh [128] proposed a three-layer trust establishment model which provides a unifying view of trust establishment, consisting of authentication process, semantic representation of trust, and trust evaluation. Shand et al. [124] introduced a trust framework in which individuals compute their trust in information by combining their own trust assumptions with others' recommendations. Bertino et al. [129] presented a trust negotiation framework, which allows entities to establish mutual trust on first contact through an exchange of digital credentials. Other trust acquirement mechanisms include verifying a requester's reputation ranked by independent third parties [67] and checking "spam request" and black listings [15]. The future work could select appropriate trust acquisition mechanisms among these existing models and integrate them into the privacy protection model proposed. But the work must bear in mind the resource-constrained environments that the Privacy Agent will play with.

- ***Conducting formal usability studies and developing actual user interface***

The privacy technology proposed in this work is a human-centric solution. Not only does the separated privacy control introduced by the distributed protection model allow multiple parties to contribute to privacy protection with a clear accountability of each party, but also various mechanisms are proposed in the Privacy Agent technology to facilitate automated processes of privacy control, and to enable relatively unobtrusive user participation in controlling the disclosure of their private information, including getting notice of relevant information disclosure, feedback, and explicit consent. These mechanisms include, but are not limited to, provisioning the privacy meta-policy that

allows users to have a high-level coarse control over information disclosure in addition to more specific and complex rules for fine-grained control, and employing ontology-based methods to perform semantic analysis of privacy policy, to automatically detect potential preference conflicts and redundancy when users update new or delete existing privacy preferences, and to perceive context before making personalized information disclosure decision.

The approach taken by the author to perform the automated process of privacy control, however, is distinguished from research efforts (such as [163]) that strived to exploit advanced machine learning techniques and sophisticated process logic to provision higher level or full machine-driven automation in the information disclosure decision making process, and attempted to make information disclosure decisions without or allowing less possible user participation. In the author's proposal, human's notice, acknowledgement, explicit consent, and feedback are staple requirements in preserving individual privacy towards dynamic context-aware computing environment where information disclosure may appear anywhere at anytime and without people knowledge. However, the balance between adopting automatic control on behalf of users and giving manual control to users over their information disclosure needs to be carefully evaluated. The experiment-based investigation by Eldin & Wagenaar [163] showed that users preferred manual privacy control to automatic control but this seemed subject to the number of times a user would tolerate the interruption.

In this regard, a significant work remaining in the future is to conduct formal usability studies on the Privacy Agent proposal. New efforts are required to develop actual user interface, deploy real applications, and get feedback from both end users and application developers, in order to understand how people perceive and use the solution in realistic situations. The salient privacy work [17, 61] provided some methods and good survey questions to carry out usability studies. A greater understanding of the general HCI requirements for privacy (e.g., how individual differences affect use) will be required when developing actual user interface [64].

Another important aspect of the usability study is the use of the Privacy Policy/Preference Language. One of important objectives of the language is to be generic, in other words, it is required to be able to state application-independent data collecting policies, and to be user-adaptable in response to the varieties of individual privacy preferences. The author has followed the widely-accepted P3P practice to construct policy elements of the language, so that data collecting policies can be represented in a standardized and machine-readable format shared by various types of applications. The development of another part of the language, i.e. the preference elements, however, is mainly based on the author's perception to privacy requirements in dynamic context-aware environments. First-hand empirical evidence, through mechanisms such as user feedback, is needed to prove the adaptability of the language. The author also considers fostering the use of the language by submitting it to relevant language standardization organization or forum. One possible place is the P3P forum. The language may, as a derivative work of the P3P standard, attract a variety of interests from industry and academy.

- ***Improving context reasoning capability***

In Chapter 6 and 7, the author demonstrated the feasibility of coupling rule-based method with the semantic web programming framework (i.e. Jena) to support context reasoning. Comparing with the conventional approaches (e.g. [36, 38]) that use programming class objects to model context and "hardwire" procedures to interpret context, the rule-based approach improves flexibility of context reasoning implementation. However, the rule-based approach has certain weaknesses. First, there exists various ways to perceive human activities; inference rules can be very complicated and might not scale well when the number of conditional factors increases in logical inference rules in order to achieve certain inference accuracy. Second, defining context inference rules often requires special knowledge of domain experts, and it might be difficult to effectively define inference rules for all possible contexts

that the Privacy Agent will use. In this work, the author only exemplified simple inference rules based on common sense to show how the rule-based application-specific logic over ontology description works.

Future work on context reasoning may include investigating the feasibility of incorporating more sophisticated logic and Artificial Intelligence techniques [164] in the context perception process, in particular, to define and apply inference rules. The work conducted by Krumm and Horvitz [165] provided such an example. It proposed a probabilistic approach to infer a person's location in indoor SmartSpace environments and to decide whether or not a person is in motion. The experiments conducted by them showed that their probabilistic reasoner is able to infer a person's location and activity with acceptable accuracy. It is also claimed that their approach is more adaptive to dynamic environments than the rule-based method, as developers of a smart space cannot always anticipate all possible changes that could occur in the space. Incorporating sophisticated machine intelligence in the context reasoning process, however, may be time-consuming. Quantitative performance analysis may be required when adapting it for critical applications and in resource-constrained platforms that might have limited processing power and memory.

- ***Continuing performance evaluation work***

As stated in the evaluation work in section 7.2.5. the author's experience with using ontology and logic inference tools during the evaluation experiments has revealed that the potential of employing semantic reasoning to conduct semantic policy analysis of privacy preferences and context perception has been limited by both immaturity of ontology standards and a lack of tool support for reasoning axiomatic rules over ontology description. It was thus suggested that a future direction for the performance evaluation may include continuing to monitor the development of ontology standards and tools, and exploring alternative evaluation setups and tools when they are available to prove the consistency of experimental results obtained using HP's Jena Semantic

Web Framework in this work. Also, the performance evaluation work on the semantic reasoning may look into some other factors that have not been addressed in this doctoral work. The processing power of the Privacy Agent in real situations is such a factor that may affect evaluation results.

Aside from the work validating the semantic reasoning approaches, there is a need to evaluate other system and operational issues associated with applying the Privacy Agent technology in practical and real deployment scenarios. An interesting issue, among others, is to measure how the response time of the Privacy Agent is effected by the application of different security mechanisms and enhanced trust models, although the author argues that the overhead caused by setting up a secure and trustful connection is not unique to the privacy protection solution proposed in this work.

References

[1] Strang, T. and Linnhoff-Popien, C., (2004) A context modeling survey. In Proc. of the First International Workshop on Advanced Context Modeling, Reasoning And Management, in the 6th International Conference on Ubiquitous Computing (UbiComp'04), Nottingham, England, September 2004, pp. 33-40

[2] Meyer, S. and Rakotonirainy, A., (2003) A survey of research on context-aware homes. In Proc. of the Australasian information security workshop conference on ACSW frontiers, Adelaide, Australia, February 2003, Vol. 21, pp.159-168

[3] Weiser, M., Gold, R., and Brown, J.S., (1999) The Origins of Ubiquitous Computing Research at PARC in the Late 1980s. *IBM Systems Journal* 1999. 38(4): pp. 693-696.

[4] Harper, R.H., (1996) Why People Do and Don't Wear Active Badges: A Case Study. In Proc. of ACM Conference on Computer Supported Cooperative Work (CSCW'96). Cambridge, MA. USA, November 1996, pp. 297-318

[5] Barkhuus, L. and Dey, A.K., (2003) Location-based services for mobile telephony: a study of users' privacy concerns. In Proc. of the 9th IFIP TC13 International Conference on Human-Computer Interaction (INTERACT'03), Zurich, Switzerland, September 2003, pp. 709-712

[6] Kaasinen, E., (2003) User Needs for Location-aware Mobile Services. *Personal and Ubiquitous Computing* 2003. 7(1): pp. 70-79

[7] Doheny-Farina, S., (1994) The Last Link: Default = Offline, or Why Ubicomp Scares Me, *Computer-mediated Communication*, Vol.1 (6): pp. 18-20, 1994.

[8] Brin, D., (1998) *The Transparent Society*. Reading, MA: Perseus Books, 1998.

[9] Garfinkel, S., (2001) *Database Nation: The Death of Privacy in the 21st Century*: O'Reilly & Associates, 2001.

[10] Sloane, L. (1992) Orwellian Dream Come True: A Badge That Pinpoints You, *New York Times* pp.14, 12th September 1992.

[11] Whalen, J., (1995) You're Not Paranoid: They Really Are Watching You, *Wired Magazine*, 3(3): pp. 95-85, 1995.

[12] Ebling, M.R., Hunt, G. D. H., and Lei, H., (2001) Issues for context services for pervasive computing. In Proc. of the Workshop on Middleware for Mobile Computing 2001, Heidelberg, Germany, November 2001

[13] Satyanarayanan, M., (2001) Pervasive computing: Vision and challenges. *IEEE Personal Communications*, 8(4):10–17, August 2001.

[14] Langheinrich, M., (2001) Privacy by design— principles of privacy-aware ubiquitous systems. In Proc. of International Conference on Ubiquitous Computing (UbiComp'01), Atlanta, GA, USA, September 2001, vol. 2201 of *Lecture Notes in Computer Science*, pp. 273–291.

[15] Myles, G., Friay, A., and Davies, N., (2003) Preserving privacy in environments with location-based applications. *IEEE Pervasive Computing*, 2(1): pp.56-64, January -March 2003

[16] Hengartner, U., and Steenkiste, P., (2003) Access control to information in pervasive computing environments. In Proc. of the 9th Workshop on Hot Topics in Operating Systems, Lihue, Hawaii, USA, May 2003, pp.17-27

[17] Gandon, F. L., and Sadeh, N. M., (2004) Semantic web technologies to reconcile privacy and context awareness. *Web Semantics Journal*, Vol.1(3), 2004

[18] Scott Lederer, I. Hong, K. Dey, A. Landay, (2004) Personal privacy through understanding and action: five pitfalls for designers, *Personal and Ubiquitous Computing*, Vol. 8 Issue 6, November 2004

[19] Zhang, N., and Todd C., (2005) A Generic Prototype Architecture Prototype for Context-Aware Systems. In Proc. of London Communications Symposium (LCS 2005), London, UK, September 2005

[20] Schilit, B.N., and Theimer, M.M.,(1994) Disseminating active map information to mobile hosts, *IEEE Network*, Vol. 8, Issue 5, pp. 22 – 32, September 1994.

[21] Brown, M., (1996) Supporting User Mobility, In Proc. of the IFIP'96 Conference on Mobile Communications, Canberra, Australia, November 1996, pp. 69-77

[22] Dey, A. K., (2001) Understanding and using context. *Journal of Personal and Ubiquitous Computing*, 2001, Vol. 5 (1), pp. 4-7

[23] Yang, K. Galis, A. Todd, C., (2003) Policy-driven Mobile Agents for

Context-aware Service in Next Generation Networks, In Proc. of the 5th international workshop on mobile agents for telecommunication agents (MATA'03), Marrakech, Morocco, October 2003, pp.111-120

[24] Pascoe, J., (1997) The Stick-e Note Architecture: extending the interface beyond the user, In Proc. of the second International Conference on Intelligent User Interfaces, Orlando, Florida, USA, January 1997, pp.261-264

[25] Pascoe, J., N. Ryan, N., and Morse, D., (1999) Issues in developing context-aware computing, In Proc. of the First International Symposium on Handheld and Ubiquitous Computing (HUC'99), Karlsruhe, Germany, June 1999, pp. 208-221

[26] Dey, A.K., Salber, D, and Abowd, G.D., (1999) The Context Toolkit: Aiding the Development of Context-Enabled Applications, In Proc. of Conference on Human Factors in Computing Systems (CHI'99), Pittsburgh, PA, USA, May 1999, pp. 431-441

[27] Brown, P.J., Bovey, J.D., Chen, X., (1997) Context-aware applications: from the laboratory to the marketplace, *IEEE Personal Communications*, Vol. 4, Issue 5, pp. 58 – 64, October 1997.

[28] Want, R., Hopper A., Falcao, V. and Gibbons, J., (1992) The active badge location systems. *ACM Transactions on Information Systems*, 10(1): pp.91-92

[29] Bennett, F., Richardson, T., and Harter A., (1994) Teleporting-making applications mobile. In Proc. of IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, California, USA, December 1994, pp. 82-84, IEEE Computer Society Press.

[30] Ashana, A., Cravatts, M., and Krzyzanowski, P., (1994) An indoor wireless system for personalized shopping assistance. In Proc. of IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, California, USA, December 1994, pp. 69-74

[31] Want, R., Schilit, B.N., Adams, N.I., (1996) The ParcTab Ubiquitous Computing Experiment. In *Mobile Computing*, Chapter 2 (pp.28-43), Tomasz, I. and Korth, H.F., Kluwer Academic Publishers, 1996

[32] Abowd, G.D., Atkeson, C.G, and Hong, J., (1997) Cyberguide: A mobile

context-aware tourguide. *Wireless Networks*, 3(5) pp.421-433, October 1997

[33] Jonsson, A., Giaffreda, R., Barachi, M., Glitho, R., Belqasmi, F., Smirnov, M., Kleis, M., Reichert, C., Karmouch, A., Khedr, M., Karlsson, A., Laamanen, H., Helin, H., Galis, A., Ocampo, R., and Zhang, J., (2005) Ambient Networks ContextWare: First Paper on Context-Aware Networks. *Deliverable Report D-6-1, Ambient Networks Project*, January 2005. Document number IST-2002-507134-AN/WP6/D61, available at http://www.ambient-networks.org/phase1web/publications/D6-1_PU.pdf

[34] CONTEXT Project website: http://context.upc.es/index.html/

[35] Giaffreda, R., Karmouch, A., Jonsson, A., Karlsson, A.M., Smirnov, M.I., Glitho, R., Galis, A., (2004) Context-aware Communication in Ambient Networks, Wireless World Research Forum (WWWRF) #11, Oslo, Norway, June 2004

[36] Schilit, B., (1995) System architecture for context-aware mobile computing, Ph.D. thesis, Columbia University, 1995, available at http://schilit.googlepages.com/schilit-thesis.pdf

[37] Castro, P., and Muntz, R., (2000) Managing context for smart spaces. *IEEE personal communications*, 7(5), pp.44-46

[38] Dey, A. K., Abowd, G.D., and Salber D., (2001) A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications, *Human-Computer Interaction Journal*, Vol. 16(2-4): pp. 97-166, 2001.

[39] Hong, J.H., (2002) The context fabric: an infrastructure for context-aware computing, In Proc. of Conference on Human Factors in Computing Systems 2002 (CHI'02), Minneapolis, Minnesota, USA, April 2002, pp. 554 – 555 (Also at *Human-Computer Interaction (HCI) Journal*, 2001. v.16 pp.2-3)

[40] Roman, M., Hess, C., Cerqueira, R., Ranganathan, A., Campbell, R.H., Nahrstedt, K., (2002) Gaia: A middleware infrastructure for active spaces. *IEEE Pervasive Computing*, Special Issue on Wearable Computing, 2002, Vol. 1, pp.74-83

[41] Chen, G., (2004) Solar: Building A Context Fusion Network for Pervasive Computing, PhD thesis, Technical Report TR2004-514, Dartmouth College, http://www.cs.dartmouth.edu/reports/abstracts/TR2004-514/

[42] Chen, G., Li, M., and Kotz, D., (2004) Design and implementation of a large-scale context fusion network. In Proc. of the 1st Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04), Boston, Massachusetts, USA, August 2004, pp. 246-255

[43] Henricksen, K., and Indulska, J., (2004) A software engineering framework for context-aware pervasive computing. In Proc. of the second IEEE International Conference on Pervasive Computing and Communications (PerCom'04), Orlando, FL, USA, March 2004, pp. 77-86, IEEE Computer Society.

[44] Henricksen, K., Indulska, J., McFadden, T., and Balasubramaniam, S., (2005) Middleware for distributed context-aware systems. In Proc. of the International Symposium on Distributed Objects and Applications (DOA'05), Agia Napa, Cyprus, November 2005, pp. 846-863

[45] Balakrishnan D., Barachi M.E., Karmouch, A., and Glitho, R., (2005) Challenges in modeling and disseminating context information in Ambient Networks. In Proc. of the Second International Workshop, MATA 2005, Montreal, Canada, October 2005, pp. 32-42

[46] Chen, H., Finin, T. and Joshi, A., (2004) Semantic web in the context broker architecture. In Proc. of the 2nd IEEE International Conference on Pervasive Computing and Communications (PerCom'04), Orlando, Florida, USA, March 2004, pp.277-286

[47] Strang, T., Linnhoff-Popien, C., and Frank, K., (2003) Integration issues of an ontology based context modeling approach. In Proc. of IADIS International Conference WWW/Internet (ICWI 2003), Algarve, Portugal, November 2003, pp.361-368

[48] Wang X. H., Zhang, D. Q., Gu, T., and Pung, H. K., (2004) Ontology based context modeling and reasoning using OWL. In Proc. of the workshop on Context Modeling and Reasoning, in the 2nd IEEE International Conference on Pervasive Computing and Communications (PerCom'04), Orlando, Florida, USA, March 2004, pp.18-22

[49] Gruber, T.R., (1993) A Translation Approach to Portable Ontology Specifications.

Knowledge Acquisition, 5(2): pp.199-220, 1993, available at http://tomgruber.org/writing/ontolingua-kaj-1993.pdf

[50] Noy, N. F., and McGuinness, D. L., (2001) Ontology Development 101: A guide to creating your first ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001, available at http://www-ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-noy-mcguinness.html

[51] McGuinness, D.L., and Harmelen, F. V., (2004) Web Ontology Language (OWL), W3C Recommendation 10 February 2004, http://www.w3.org/2004/OWL/

[52] Rule Markup Language (RuleML), http://www.ruleml.org/

[53] Henricksen, K., Livingstone, S., and Indulska, J., (2004) Towards a hybrid approach to context modeling, reasoning and interoperation. In Proc. of the First International Workshop on Advanced Context Modeling, Reasoning And Management, in the 6th International Conference on Ubiquitous Computing (UbiComp'04), Nottingham, England, September 2004, pp. 54-61

[54] Warren S., and Brandeis L., (1890) The right to privacy. *Harvard Law Review*, 4: 193-220, 1890, available at http://www.lawrence.edu/fac/boardmaw/Privacy_brand_warr2.html

[55] Langheinrich, M., (2005) Personal Privacy in Ubiquitous Computing – Tools and System Support, PhD thesis No. 16100, ETH Zurich, Zurich, Switzerland, May 2005

[56] Westin, A., (1967) *Privacy and Freedom*, Atheneum, New York

[57] Hong, J.I., (2005) An architecture for privacy-sensitive ubiquitous computing, PhD thesis, University of California, Berkley, available at http://www.cs.cmu.edu/~jasonh/publications/jihdiss.pdf

[58] Westin, A., (1995) Privacy in America: an historical and sociopolitical analysis. In Proc. of the national privacy and public policy symposium, Hartford, Connecticut, November 1995

[59] Cranor, L.F., Reagle, J., and Ackerman, M.S., (1999) Beyond concern:

understanding net users' attitudes about online privacy. Technical Report TR 99.4.3, AT&T Labs-Research, April 1999.

[60] Henricksen, K., Wishart, R., McFadden, T., Indulska, J., (2005) Extending context models for privacy in pervasive computing environments. In Proc. of the 2nd International Workshop on Context Modeling and Reasoning (CoMoRea), PerCom'05 Workshop, Sheraton-Kauai Resort, Kauai, Hawaii, USA, March 2005, IEEE Computer Society, pp.20–24

[61] Hong, J.I., and Landay, J.A., (2004) An architecture for privacy sensitive ubiquitous computing. In Proc. of the 2nd international conference on mobile systems, applications and services (MobiSYS '04), Boston, Massachusetts, USA, June 2004, pp. 177–189. ACM Press, 2004.

[62] Kobsa, A., and Schreck, J.O., (2003) Privacy Through Pseudonymity in User-Adaptive Systems, ACM Transactions on Internet Technology, Vol. 3, No. 2, May 2003, pp. 149–183.

[63] Lessig, L., (1999) *Code and Other Laws of Cyberspace*. 1999, New York NY: Basic Books.

[64] Ackerman, M.S., Darrell, T., and Weitzner, D.J., (2001) Privacy in Context. *Human-Computer Interaction* 2001, 16(2-4), pp. 167-176.

[65] Pfitzmann, A., and Koehntopp, M., (2001) Anonymity, unobservability, and pseudonymity – a proposal for terminology. In Proc. of Workshop on Design Issues in Anonymity and Unobservability, vol.2009 of LNCS. Springer-Verlag, 2001, pp. 1-9

[66] Seigneur, J. M., and Jensen, C. D., (2004) Trust enhanced ubiquitous payment without too much privacy loss, In Proc. of the 2004 ACM symposium on Applied computing, Nicosia, Cyprus, pp. 1593 - 1599

[67] Miranda, H., and Rodrigues, L., (2006) A Framework to Provide Anonymity in Reputation Systems. In Proc. of the 3rd Annual International Conference on Mobile and Ubiquitous Systems - Workshops , pp. 1-4

[68] Spreitzer, M., and Theimer, M., (1993) Providing location information in a ubiquitous computing environment (panel session). In Proc. of the Fourteenth

ACM Symposium on Operating Systems Principles (SOSP '93), pp. 270–283, ACM Press, 1993.

[69] Beresford A. R., and Stajano F., (2004) Mix zones: User privacy in location-aware services. In Proc. of the First IEEE International Workshop on Pervasive Computing and Communication Security (PerSec'04), Orlando, Florida, March 2004, pp.127-131 Available at http://www.cl.cam.ac.uk/~fms27/papers/2004-BeresfordSta-mix.pdf

[70] Gruteser M. and Grunwald. D., (2003) Anonymous usage of location-based services through spatial and temporal cloaking. In Proc. of the First International Conference on Mobile Systems, Applications, and Services (MobiSys'03), San Francisco, USA, May 2003, pp. 31-42, ACM USENIX.

[71] Chaum, D., (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, Vol.24, issue 2, pp.84–90, February 1981. Available at http://world.std.com/~franl/crypto/chaum-acm-1981.html

[72] Gorlach, A., Heinemann, A., and Terpstra, W.W., (2004) Survey on Location Privacy in Pervasive Computing, In Proc. of Workshop on Security and Privacy in Pervasive Computing 2004, The Springer International Series in Engineering and Computer Science, Vol. 780 pp. 23-34

[73] EC – European Commission. Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (directive on privacy and electronic communications). *Official Journal of the European Communities*, L 201:37–47, July 2002 http://europa.eu.int/smartapi/cgi/sga_doc?smartapi!celexapi!prod!CELEXnumdoc&lg=en&numdoc=32002L0058&model=guichett

[74] Sweeney, L., (2002) k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5): pp.557–570, 2002. http://privacy.cs.cmu.edu/people/sweeney/kanonymity.pdf

[75] Hull, R., Kumar, B., Lieuwen D., and Patel-Schneider, P., (2004) Enabling Context-Aware and Privacy-Conscious User Data Sharing, In Proc. of the IEEE

International Conference on Mobile Data Management (MDM '04), Berkeley, California, USA, January 2004, pp. 187-198

[76] Cuellar, J., Morris J., Mulligan, D., Peterson, J., and Polk J., (2004) Geopriv Requirement, RFC 3693, IETF February 2004, http://www.ietf.org/rfc/rfc3693.tex

[77] Hengartner, U., and Steenkiste, P., (2005) Access Control to People Location Information, ACM Transactions on Information and System Security, Vol. 8, No. 4, November 2005, pp. 424–456

[78] Zhang, G., and Parashar, M., (2004) Context-aware Dynamic Access control for pervasive applications. In Proc. of the Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2004), 2004 Western MultiConference (WMC), San Diego, CA, USA, January 2004, available at http://www.caip.rutgers.edu/TASSL/Papers/automate-sesame-cnds-04.pdf

[79] Covington, M. J., Long, W., Srinivasan, S., Dey, A., Ahamad, M., and Abowd, G., (2001) Securing Context-Aware Applications Using Environment Roles. In Proc. of the 6th ACM Symposium on Access Control Models and Technologies (SACMAT '01), Litton-TASC, Chantilly, Virginia, USA, May 2001, pp. 10–20

[80] Neumann, G. and Strembeck, M., (2003) An Approach to Engineer and Enforce Context Constraints in an RBAC Environment. In Proc. of the 8th ACM Symposium on Access Control Models and Technologies (SACMAT'03), Villa Gallia, Como, Italy, June 2003, pp. 65–79

[81] Huang, X., Wang, H., Chen Z., and Lin, J., (2006) A Context, Rule and Role-Based Access Control Model In Enterprise Pervasive Computing Environment, In proc. of The 1st International Symposium on Pervasive Computing and Applications, 3-5 August, 2006, pp. 497-502

[82] Guo, Y., Hong, F., Zhang, Q., and Li, R., (2005) An Access Control Model for Ubiquitous Computing Application. In Proc. of the 2nd International Conference on Mobile Technology, Applications and Systems (IEEE Mobility Conference 2005), Guangzhou, China, November 2005, pp.1 – 6

[83] AURA project: Distraction-free ubiquitous computing, http://www.cs.cmu.edu/~aura/

[84] Al-Muhtadi, J., Ranganathan, A., Campbell, R., and Mickunas, M. D., (2003) Cerberus: A Context-Aware Security Scheme for Smart Spaces. In Proc. of the IEEE International Conference on Pervasive Computing and Communications (PerCom'03), Dallas-Fort Worth, Texas, USA ,March 2003, pp. 489–496

[85] Parducci, B., Lockhart, H., Mishra, M., Levinson, R., Clark, J.B., and Kumar, R., OASIS eXtensible Access Control Mark-up Language (XACML), http://www.oasis-open.org/committees/xacml/

[86] Minami, K., and Kotz, D., (2004) Secure context-sensitive authorization, In Proc. of the 3rd IEEE International Conference on Pervasive Computing and Communications (PerCom'05), Sheraton-Kauai Resort, Kauai, Hawaii, USA March 2005, pp. 257–268. Also available at Technical Report TR2004-529, Dartmouth College, http://www.cs.dartmouth.edu/reports/abstracts/TR2004-529/

[87] Stamp, M., (2005) *Information Security: Principles and Practice.* Wiley 2005, ISBN: 0471738484

[88] Hes, R., and Borking J., (2000) *Privacy-Enhancing Technologies: The Path to Anonymity.* Ontario Information and Privacy Commissioner, Canada and Registratiekamer, Netherlands. Revised edition (2000). ISBN: 90747087124

[89] Berglund, A., Scott Boag, S., Chamberlin, D., Fernandez, M.F., Kay, M., Robie, J., and Simeon, J., (2007) XML Path Language (XPath 2.0). W3C Recommendation, 23 January 2007, http://www.w3.org/TR/xpath20/

[90] Freier, A.O., Karlton, P., and Kocher, P.C., (1996) The SSL protocol (version 3.0.) Internet-draft, IETF, November 1996.

[91] Hengartner, U., and Steenkiste, P., (2005) Exploiting Hierarchical Identity-Based Encryption for Access Control to Pervasive Computing Information, In Proc. of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SecureComm'05), Athens, Greece, September 2005, pp. 384 - 396

[92] Ren, K., and Lou W., (2007) Privacy-enhanced, attack-resilient access control in pervasive computing environments with optional context authentication capability. *Mobile Networks and Applications,* Volume 12 , Issue 1, February 2007, pp 79-92

[93] Jiang, X. and Landay, J.A., (2002) Modeling Privacy Control in Context-Aware Systems IEEE Pervasive Computing, July 2002 Vol. 1 Issue 3

[94] Zuidweg, M., (2003) A P3P-based privacy architecture for a context-aware services platform, Master Thesis, August 2003, University of Twente, available at http://asna.ewi.utwente.nl/education/Student%20assignments/completed%20bachelor%20and%20master%20assignments/ARCH-2003-07.pdf

[95] Cranor, L., Dobbs, B., Egelman, S., Hogben, G., and Schunter, M., The Platform for Privacy Preferences 1.1 (P3P1.1), http://www.w3.org/P3P/, July 2005

[96] Ashley, P., Hada, S., Karjoth, G., Powers, C., and Schunter M., (2003) Enterprise Privacy Authorization Language (EPAL). IBM Research Report RZ 3485 (# 93951) 03/03/2003, IBM Zurich Research Laboratory, Available at http://www.zurich.ibm.com/security/enterprise-privacy/epal/

[97] Hengartner, U., and Steenkiste, P., (2006) Avoiding Privacy Violations Caused by Context-Sensitive Services. *Elsevier Journal of Pervasive and Mobile Computing (PMC)*, PerCom2006 special issue, 2(4): pp. 427–452, November 2006

[98] Hengartner, U., and Steenkiste, P., (2007) Distributed, uncertainty-aware access control for pervasive computing. In Proc. of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops (PerComW'07), White Plains, New York, USA, March 2007, pp.241-246

[99] Dockhorn C. P., (2003) Towards a Services Platform for Context-aware Applications, Master Thesis, August 2003, University of Twente, available at http://wwwhome.cs.utwente.nl/~dockhorn/files/thesis_dockhorn.pdf or https://doc.freeband.nl/dsweb/View/Collection-5649

[100] Imamura, T., Dillaway, B., Simon, E., (2002) XML Encryption Syntax and Processing Candidate Recommendation, http://www.w3.org/TR/2002/CR-xmlenc-core-20020304/

[101] Wishart, R., Henricksen, K., and Indulska, J., (2005) Context obfuscation for privacy via ontological descriptions. In Proc. of the 1st International Workshop on Location- and Context- Awareness (LoCA'05), Munich, Germany, May 2005, Lecture Notes in Computer Science, Springer 2005, Vol. 1678, pp. 276–288

[102] Chen, H., (2004) An Intelligent Broker Architecture for Pervasive Context-Aware Systems, PhD Thesis, University of Maryland, Baltimore County, December 14, 2004, available at http://ebiquity.umbc.edu/paper/html/id/212/An-Intelligent-Broker-Architecture-for-Pervasive-Context-Aware-Systems

[103] Chen, H., Perich, F., Finin, T., and Joshi, A., (2004) SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications. In Proc. of the 1st Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04), Boston, Massachusetts, USA, August 2004, available at http://ebiquity.umbc.edu/paper/html/id/168/

[104] Sacramento, V. Endler, M. and Nascimento, F.N. , (2005) A Privacy Service for Context-aware Mobile Computing, In Proc. of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks (Securecomm'05), Athens, Greece, September 2005, pp. 182- 193

[105] Cranor, L., Langheinrich, M., and Marchiori, M., (2001) A P3P preference exchange language 1.0 (APPEL1.0). W3C Working Draft, April 2001, http://www.w3.org/TR/P3P-preferences

[106] Agrawal, R., Kiernan, J., Srikant, R., and Xu, Y.R., (2003) An XPath-based preference language for P3P. In Proc. of the 12th International World Wide Web Conference (WWW2003), Budapest, Hungary, May 2003, pp. 629-639

[107] Ackerman, M. S., (2004) Privacy in pervasive environments: next generation labeling protocols. *Personal and Ubiquitous Computing* (November 2004) vol.8, issue 6, pp.430-439

[108] Brar, A., and Kay, J., (2005) Privacy and Security in Ubiquitous Personalized Application. In Proc. of the User Modeling Workshop on Privacy-Enhanced Personalization, July 2005, Edinburgh, UK, available at http://www.isr.uci.edu/pep05/papers/PEPp.pdf

[109] Thomas, R.K. and Sandhu, R., (2004) Models, protocols, and architectures for secure pervasive computing: challenges and research directions. In Proc. of the 2nd IEEE International Conference on Pervasive Computing and Communications

(PerCom'04) workshop, Orlando, Florida, USA, March 2004, pp.164 - 168

[110] U.S. Privacy Act of 1974, available at http://epic.org/privacy/1974act/

[111] EC – European Commission. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. *Official Journal of the European Communities*, L 281(395L0046): pp.31–50, November 1995. http://europa.eu.int/smartapi/cgi/sga_doc?smartapi!celexapi!prod!CELEXnumdoc&lg=EN&numdoc=31995L0046&model=guichett

[112] Privacy Rights Clearinghouse. A review of the fair information principles: The foundation of privacy public policy, posted on October 1997, revised on February 2004, available at http://www.privacyrights.org/ar/fairinfo.htm

[113] Organisation for Economic Co-operation and Development (OECD), OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data, 1980 September, available at http://www.oecd.org/document/0,2340,en_2649_34255_1815186_1_1_1_1,00.html

[114] Solove, D.J., and Rotenberg, M., (2003) *Information Privacy Law*, Aspen Publishers, New York, USA, 2003

[115] Wagealla, W., Terzis, S., and English, C., (2003) Trust-based model for privacy control in context aware systems. In Proc. of the Second Workshop on Security in Ubiquitous Computing at the Fifth Annual Conference on Ubiquitous Computing (UbiComp'03), Washington, USA, October 2003, available at http://www.vs.inf.ethz.ch/events/ubicomp2003sec/papers/secubi03_p03.pdf

[116] Roussaki, I., Strimpakou, M., Pils, C., Kalatzis, N., Neubauer, M., Hauser, C., and Anagnostou, M., (2006) Privacy-Aware Modelling and Distribution of Context Information in Pervasive Service Provision, Pervasive Services. In Proc. of the ACS/IEEE International Conference on Pervasive Services (ICPS'06), Lyon, France, June 2006, pp.150 - 160

[117] Blaze, M., Feigenbaum, J., and Lacy, J., (1996) Decentralized trust management,

In Proc. of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, May 1996, pp. 164–173

[118] Wang, P., and Zhang, Z.L., (2005) A computation trust model with trust network in multi-agent systems. In Proc. of the 2005 International Conference on Active Media Technology (AMT'05), Takamatsu, Kagawa, Japan, May 2005, pp.389-392

[119] Kagal, L., Finin, T., and Joshi, A., (2001) Trust-based security in pervasive computing environments. *IEEE Computer*, 34(12): pp.154–157, December 2001

[120] Chen, Z.X., Ge, L.S., Wang, H.Y., Huang, X.Z., and Lin, J.J., (2006) A Trust-Based Service Evaluation and Selection Model in Pervasive Computing Environment. In Proc. of the 1st International Symposium on Pervasive Computing and Applications, Xinjiang, China, August 2006, pp.641 - 646

[121] Liu, Z.Y., and Xiu, D.X., (2005) Agent-based automated trust negotiation for pervasive computing. In Proc. of the Second International Conference on Embedded Software and Systems, Xian, China, December 2005, pp. 300-309

[122] Shankar N., and Arbaugh, W.A., (2002) On trust for ubiquitous computing. In Proc. of the workshop on Security in Ubiquitous Computing in the International Conference on Ubiquitous Computing (UbiComp'02), Goteborg, Sweden, September 2002, available at http://www.teco.edu/~philip/ubicomp2002ws/organize/maryland.pdf

[123] Yuan, W.W., Guan, D.H., Lee, S.Y., and Lee, Y.K., (2006) A Context-Based Architecture for Reliable Trust Model in Ubiquitous Environments. In Proc. of the 14th IEEE International Conference on Networks (ICON'06), Singapore, September 2006 Vol. 1, pp.1 – 5

[124] Shand, B., Dimmock, N., and Bacon, J., (2003) Trust for ubiquitous, transparent collaboration. In Proc. of the First IEEE Conference on Pervasive Computing and Communications (PerCom'03), Dallas-Fort Worth, Texas, USA, March 2003, pp. 153–160

[125] Westerinen, A., Strassner, J., Scherling, M., Quinn, B., Herzog, S., Huynh, A., Carlson, M., Perry, J., and Waldbusser, S., (2001) Terminology for Policy-Based

Management. IETF RFC3198, available at http://www.ietf.org/rfc/rfc3198.txt

[126] Bartel, M., Boyer, J., Eastlake, D., Fox, B., LaMacchia, B., Reagle, J., Simon, E., and Solo, D., (2002) XML signature syntax and processing. W3C Recommendation, World Wide Web Consortium (W3C), February 2002 http://www.w3.org/TR/xmldsig-core/

[127] Mazzuca, P., (2004) Access Control in a Distributed Decentralized Network: An XML Approach to Network Security using XACML and SAML, Technical Report TR2004-506, Dartmouth College, http://www.cs.dartmouth.edu/reports/abstracts/TR2004-506/

[128] Seng, C.Y., and Arbaugh, W.A., (2006) A Secure Trust Establishment Model. In Proc. of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06), Taichung, Taiwan, June 2006, pp.78 - 85

[129] Bertino, E., Ferrari, E., and Squicciarini, A. C., (2003) Trust-χ: An XML framework for trust negotiations. In Proc. of the seventh IFIP TC6 TC11 conference on Communications and Multimedia Security (CMS'03), Torino, Italy, October 2003, pp.146–157

[130] Cantor, S., Kemp, J., Philpott, R., and Maler, E.,(2004) Security Assertion Markup Language (SAML) v.2.0, August 2004, http://xml.coverpages.org/saml.html

[131] Hightower, J., and Boriello, G., (2001) Location Systems for Ubiquitous Computing, *IEEE Computer*, August 2001, vol. 34, no. 8, pp. 57-66

[132] Brickley, D., and Guha, R.V.,(2004) Resource Description Framework Schema (RDFS), February 2004, http://www.w3.org/TR/rdf-schema/

[133] The Protégé Ontology Editor and Knowledge Acquisition System, Stanford University, http://protege.stanford.edu/

[134] Altova SemanticWorks, http://www.Altova.com/SemanticWorks

[135] Hobbs, J. R., (2002) A DAML ontology of time, available at http://www.cs.rochester.edu/_ferguson/daml/daml-time-20020830.txt

[136] Pan, F., and Hobbs, J, R., (2004) Time in OWL-S. In Proc of AAAI-04 Spring Symposium on Semantic Web Services, Stanford University, California, 2004, available at http://www.isi.edu/~hobbs/time/pub/pan-hobbs-AAAI-SSS04.pdf

[137] Cox, S., Daisey, P., Lake, R., Portele, C., and Whiteside A., (2003) Geography markup language (gml 3.0). In *OpenGIS Documents*. OpenGIS Consortium, 2003

[138] Shehzad, A., Ngo, H., Pham, K., and Lee, S., (2004) Formal Modeling in Context Aware Systems. In Proc. of the 1st International Workshop on Modeling and Retrieval of Context (MRC 2004), Ulm, Germany, September 2004, available at http://uclab.khu.ac.kr/resources/publication/C_28.pdf

[139] Giaffreda, R., Dang, J., Glitho, R., Barachi, M. E., Belqasmi, F., Mattam, J., Kanter, T., Reichert, C., Smirnov, M., Karmouch, A., Balakrishnan, D., Harroud, H., Karlsson, A., Laamanen, H., Laukkanen, M., Ocampo, R., Jean, K., and Galis A., (2005) Ambient Networks ContextWare: Second Paper on Context-Aware Networks. Deliverable Report D-6-3, Ambient Networks Project, December 2005. Document number IST-2002-507134-AN/WP6/D63. available at http://www.ambient-networks.org/phase1web/publications/D6_3_Ambient_Networks_ContextWare_Second_Paper_on_Context-Aware_Networks_PU.pdf

[140] Lupu, E. C., and Sloman, M., (1999) Conflicts in policy-based distributed system management. IEEE Transaction on Software Engineering, Vol. 25, No. 6, 1999

[141] Jena Semantic Web Framework, http://jena.sourceforge.net/

[142] Dunlop, N., Indulska J., and Raymond K., (2003) Methods for Conflict Resolution in Policy-based Management System. In Proc. of the Seventh IEEE International Enterprise Distributed Object Computing Conference (EDOC'03), Brisbane, Australia, September 2003, pp. 98-109

[143] Tonti, G., Bradshaw, J. M., Jeffers, R., Montanari, R.., Suri, N., and Uszok, A., (2003) Semantic web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder, In Proc. of the 2nd International Semantic Web Conference (ISWC'03), Sanibel Island, Florida, USA, October 2003, pp. 419-437

[144] Kagal, L., Finin, T., and Joshi, A.,(2003) A Policy Language for A Pervasive Computing Environment, In Proc. of the IEEE 4th International Workshop on Policies for Distributed Systems and Networks (Policy'03), Lake Como, Italy, June 2003, pp. 63-74

[145] Uszok, A., Bradshaw, J., Jeffers, R., Suri, N., Hayes, P., Breedy, M., Bunch, L., Johnson, M., Kulkarni, S., and Lott, J., (2003) KAoS Policy and Domain Services: Toward a Description-Logic Approach to Policy Representation, Deconfliction, and Enforcement. In Proc. of IEEE 4th International Workshop on Policies for Distributed Systems and Networks (Policy'03), Lake Como, Italy, June 2003, pp.93-96

[146] Damianou, N., Dulay, N., Lupu, E., and Sloman, M., (2001) The Ponder Policy Specification Language. In Proc. of Workshop on Policies for Distributed Systems and Networks (Policy'01), Bristol, UK, January 2001, pp.17-28

[147] Kolari P., (2004) Enhancing Web Privacy with Policy Language and Trust, Master Thesis, 2004, University of Maryland, Baltimore County, May 10, 2004, available at http://ebiquity.umbc.edu/paper/html/id/195/Enhancing-Web-Privacy-with-Policy-Language-and-Trust

[148] Brickley D., and Miller, L., FOAF (Friends-of-A-Friends ontology) Vocabulary Specification v.0.9, May 2007, http://xmlns.com/foaf/0.1/

[149] Khedr, M., and Karmouch, A., (2003) Exploiting Agents and SIP for Smart Context Level Agreements. In Proc. of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM'03), Victoria, B.C., Canada, Vol. 2, August 2003, pp. 522–525

[150] Serrano, J., Serrat, J., and Galis, A., (2006) Ontology-Based Context Information Modelling for Managing Pervasive Applications. In Proc. of the International Conference on Autonomous and Autonomic Systems (ICAS'06), San Jose, CA, USA, July 2006, pp. 47–53

[151] Simple Object Access Protocol (SOAP), http://www.w3.org/TR/soap12-part1/

[152] JAVA Remote Method Invocation (RMI), http://java.sun.com/products/jdk/rmi/

[153] OMG Common Object Request Broker Architecture (CORBA), http://www.omg.org/corba/

[154] Apache AXIS2, http://ws.apache.org/axis2/

[155] Apache Tomcat, http://tomcat.apache.org/

[156] WS-Security, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss

[157] Apache Xindice, http://xml.apache.org/xindice/

[158] Resource Description Framework (RDF), http://www.w3.org/RDF

[159] SPARQL Protocol and RDF Query Language (SPARQL), http://www.w3.org/TR/rdf-sparql-query

[160] JESS, the Rule Engine for the Java platform, http://herzberg.ca.sandia.gov/

[161] Pellet OWL reasoner, http://pellet.owldl.com/

[162] Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., and Dean., M., (2004) SWRL: A Semantic Web Rule Language Combining OWL and Rule, May 2004, http://www.w3.org/Submission/SWRL/

[163] Eldin, A.A. and Wagenaar, R., (2004) Towards users driven privacy control. In Proc. of 2004 IEEE International Conference on Systems, Man and Cybernetics, Hague, Netherlands, October 2004, Volume 5, pp. 4673 – 4679

[164] Akman, V., Bouquet, P., Thomason, R., and Young, R.A. (Eds.) (2001), Modeling and Using Context, In Proc. of the Third International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT' 01), Dundee, UK, July 2001, Lecture Notes in Computer Science , Vol. 2116

[165] Krumm, J., and Horvitz, E., (2004) LOCADIO: Inferring motion and location from Wi-Fi signal strengths. In Proc. of the 1st Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04), Boston, Massachusetts, USA, August 2004, pp.4-13

[166] Horridge, M., Knublauch, H., Moulton, G., Rector, A., Stevens, R., and Wroe, C., (2004) A Practical Guide To Building OWL Ontologies Using Protégé and CO-ODE Tools (edition 1.0). University of Manchester and Stanford University, August 27 2004, available at http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf

[167] Reynolds, D., (2007) Jena 2 Inference Support Document (v.1.37). last modified in September 2007, available at http://jena.sourceforge.net/inference/index.html

[168] Deng, X., Haarslev, V., and Shiri, N., (2005) A Framework for Explaining

Reasoning in Description Logics. In Proc. of the International Symposium on Explanation-aware Computing, AAAI Fall Symposium 2005, Washington DC, USA, November 2005, available at http://www.cs.concordia.ca/~haarslev/publications/AAAI-FSS-2005.pdf

[169] Kay, M., (2007) XSL Transformations (XSLT) Version 2.0, W3C Recommendation, 23 January 2007, http://www.w3.org/TR/xslt20/

[170] Ocampo, R., (2006) Understanding, Modelling & Using Flow Context. PhD Thesis, University College London, E&EE Department, London UK, December 2006

[171] Schilit, B.N., Theimer, M.M., and Welch, B.B., (1993) Customizing mobile applications, In Proc. of USENIX Mobile and Location-Independent Computing Symposium, Cambridge, Massachusetts, USA, August 1993, pp.129-138

[172] Voelker, G., and Bershad, B., (1994) Mobisaic: An information system for mobile wireless computing environment. In Proc. of IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, California, USA, December 1994, pp.185-190

[173] Samulowitz, M., Michahelles, F. and Linnhoff-Popien, C., (2001) CAPEUS: An architecture for context-aware selection and execution of services. In Proc. of the IFIP TC6/WG6.1 Third International Working Conference on Distributed Applications and Interoperable Systems, Kraków, Poland, September 2001, pp. 23-40

[174] Klyne, G., Reynolds, F., Woodrow, C., and Ohto, H., (2001) Composite Capabilities/Preferences Profile (CC/PP): Structure and Vocabularies, W3C Working Draft, 2001, http://www.w3.org/Mobile/CCPP/

[175] Wireless Application Protocol Forum, Ltd., (2001), User Agent Profile (UA Prof), Version 20-October- 2001 http://www.openmobilealliance.org/tech/affiliates/wap/wap-248-uaprof-20011020-a.pdf

[176] Held, A., Buchholz, S., and Schill, A., (2002) Modeling of context information for pervasive computing applications. In Proc. of the 6th World Multiconference on

Systemics, Cybernetics, and Informatics (SCI'02), Orlando, Florida, USA, July 2002, available at http://citeseer.ist.psu.edu/held02modeling.html

[177] Chtcherbina, E., and Franz, M., (2003) Peer-to-peer coordination framework (p2pc): Enabler of mobile ad-hoc networking for medicine, business, and entertainment. In Proc. of International Conference on Advances in Infrastructure for Electronic Business, Education, Science, Medicine, and Mobile Technologies on the Internet (SSGRR2003), L'Aquila, Italy, January 2003

[178] Bauer, J., (2003) Identification and modeling of contexts for different information scenarios in air traffic. Diplomarbeit, Technische Universit"at Berlin, March 2003, available at http://talis.eurocontrol.fr/pub/thesis-030327%20Indentif-and-modeling-of-contexts-in-ATC.pdf

[179] Henricksen, K., Indulska, J., and Rakotonirainy, A., (2002) Modeling context information in pervasive computing systems. In Proc. of the 1st International Conference on Pervasive Computing, Zurich, Switzerland, August 2002, pp.167-180

[180] Schmidt, A., Beigl, M., and Gellersen, H.W., (1999) There is more to context than location. *Computers and Graphics* 23, 6(1999), pp.893-901

[181] Schmidt, A., and Laerhoven, K. V., (2001) How to build smart appliances, *IEEE Personal Communications* (2001), pp.66-71

[182] Davies, N., Cheverst, K., Mitchell, K. and Friday, A., (1999) Caches in the air: Disseminating tourist information in the GUIDE system. In Proc. of the second IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, Louisiana, USA. February 1999, pp.11-19

[183] Gray, P., and Salber, D., (2001) Modelling and using sensed context information in the design of interactive applications. In Proc. of the 8th International Conference on Engineering for Human-Computer Interaction, Toronto, Canada, May 2001, pp.317-336

[184] Cranor, L.F., and Jr. Reagle, J., (1997) Designing a social protocol: Lessons learned from the platform for privacy preferences project. In Proc. of the

Telecommunications Policy Research Conference, Alexandria, USA, September 1997, available at http://lorrie.cranor.org/pubs/dsp/dsp.html

[185]IBM Corporation, (1999), IBM Multi-National Consumer Privacy Study. IBM Global Services, IBM Corporation. Yorktown Heights, NY.

[186]Market & Opinion Research International, (2003) Privacy and Data-Sharing -Survey of Public Awareness and Perceptions, available at http://www.dca.gov.uk/majrep/rights/mori-survey.pdf

List of Publications

1. **Zhang, N.**, and Todd, C., (2007) An Ontology-Based Solution for Privacy Policy/Preference Expression towards Context-Aware Pervasive Computing Environments. In Proc. of the 4th International Workshop on Managing Ubiquitous Communications and Services (MUCS 2007), May 2007
2. **Zhang, N.**, and Todd, C., (2006) A Privacy-Respecting Context-Aware Architecture. In Proc. of the IET International Conference on Wireless, Mobile & Multimedia Networks, China, November 2006
3. **Zhang, N.**, and Todd, C., (2006) A Privacy Agent in Context-Aware Ubiquitous Computing Environments. In Proc. of the 10th IFIP International Conference on Communications and Multimedia Security (CMS2006), Heraklion, Crete, Greece, October 2006, LNCS 4237, pp.196-205
4. **Zhang, N.**, and Todd C., (2006) Developing a Privacy Ontology for Privacy Control in Context-Aware Systems. In Proc. of London Communications Symposium (LCS 2006), London, UK, September 2006
5. **Zhang, N.**, and Todd C., (2005) A Generic Prototype Architecture Prototype for Context-Aware Systems. In Proc. of London Communications Symposium (LCS 2005), London, UK, September 2005
6. Jonsson, A., Giaffreda, R., Barachi, M., Glitho, R., Belqasmi, F., Smirnov, M., Kleis, M., Reichert, C., Karmouch, A., Khedr, M., Karlsson, A., Laamanen, H., Helin, H., Galis, A., Ocampo, R., and **Zhang, J.**, (2005) Ambient Networks ContextWare: First Paper on Context-Aware Networks. *Deliverable Report D-6-1, Ambient Networks Project*, January 2005. Document number IST-2002-507134-AN/WP6/D61, available at http://www.ambient-networks.org/phase1web/publications/D6-1_PU.pdf
7. Giaffreda, R., Barachi, M., Glitho, R., Jonsson, A., Smirnov, M., Kleis, M., Karmouch, A., Khedr, M., Karlsson, A., Laamanen, H., Helin, H., Galis, A.,

Ocampo, R., and **Zhang, J.**, (2004) Context Information Sources and Destinations and Relationship to AN Activities. *Internal Report R-6-2, Ambient Networks Project*, October 2004. Document number IST-2002-507134-AN/WP6/R6-2. available at https://bscw.ambient-networks.org/bscw/bscw.cgi/0/7751 ; accessible to Ambient Networks project partners only

Appendix A: An overview of context information modelling techniques

Strang and Linnhoff-Popien [1] classified various context modeling approaches, based on the data structures that are used to express and exchange contexts, into key-value models, markup-based models, graphical models, object-oriented models, logic-based models and ontology-based models. This section presents the author's understanding of these existing context modelling techniques, and examines briefly their merits and shortcomings.

- ***Key-Value Model***

Key-value pair model is the simplest data structure used to model contextual information. As its name suggests, it uses a variable acting as the *key*, and the value of the variable holding actual context data. The key-value modeling often involves three basic concepts ——*Entity*, *Attributes* and a set of *Relationships* between entities —— to represent context information. *Entities* are simply people, places, and things. They represent the base level of context, on top of which more sophisticated representations can be built. *Attributes* describe some property of an entity. For example, people, places, and things all have names. *Relationships* are special kinds of attributes that point to other entities. For example, a person could currently be in a specific place, and this place could contain several things. Putting together, an *Entity* is composed of a set of intrinsic *Attributes* that define the entity itself, plus a set of *Relationships* with other entities.

Key-value pair context model was first proposed by Schilit et al. [171] in 1993 and has often adopted by context-awareness work such as Mobisaic [172] and Capeus [173]. This approach enables efficient and scalable information management, allowing for pairs recursion and pattern-matching queries. The simplicity of the key-value

approach is desirable from the management and error risk perspective, but it is also a drawback due to the lack of capabilities for sophisticated structuring to enable efficient context reasoning algorithms.

- ***Markup scheme models***

Context modeling through a markup scheme was the earliest attempt to follow a formal way to standardize the representation of context. Context-awareness researches that mainly consider context information such as persons or devices profiles often take on this approach, since the descriptive markup language naturally reflects the structure of profile information.

The markup scheme is usually based on a derivative of Standard Generic Markup Language (SGML), the superclass of all markup languages such as the popular XML A representative is Composite Capabilities/Preference Profiles (CC/PP) [174], developed by World Wide Web Consortium (W3C) for describing device capabilities and preferences with a focus on wireless devices, such as PDAs and mobile phones. It is in turn based upon the Resource Description Framework (RDF) [158], which is a technique for representing knowledge. CC/PP uses the XML serialization of RDF, one of the many ways supported by RDF to implement XML. Although CC/PP is designed to describe information about device hardware and software capabilities, it can describe a wider variety of context information as long as the context information can be described in terms of CC/PP components and attributes (or subtypes of them). Other similar markup scheme models include: User Agent Profile (UAProf) [175] standards, Comprehensive Structured Context Profiles (CSCP) [176], Pervasive Profile Description Language (PPDL) [177]. The salient context-awareness work adapting the markup scheme is represented by Stick-e notes [24] by Pascoe et al.

Common to all markup scheme modeling approaches is a hierarchical data structure consisting of markup tags with attributes and content. Since the content of the markup

tags is recursively defined by other markup tags, associated markup query languages enable efficient filtering and fast information retrieval of tags' content. In addition, the existence of scheme definitions and validation tools enable model checking and partial validation. However, markup scheme modeling approaches show deficiencies in terms of reasoning capabilities and model interoperability. They are often either proprietary or limited to a small set of contextual aspects, or both. The note-tags of the stick-e notes systems [24] are a good example of this kind of limitation.

- ***Graphical* models**

Graphical tools have been used in many places for human structuring purposes, to facilitate the understanding of intricacy of relations among different pieces of information or objects. A very well known general purpose modeling instrument is the Unified Modeling Language (UML) which has a strong graphical component (UML diagrams). Due to its generic structure, UML can also be used to model the context information. This is shown for instance by Bauer in [178], where contextual aspects relevant to air traffic management are modeled as UML extensions.

Another and more relevant example is a nicely designed graphics-oriented context model introduced in [179] by Henricksen et al. It proposes a graphical notation for describing types of context information, their sources, dependencies and constraints, as well as quality labels indicating quality of information.

The strength of the graph-based modeling approaches is directed towards structure level. They are mainly used to describe the structure of contextual knowledge and derive some codes (as in Bauer's example [178]) or an Entity-Relationship Model (as in Henricksen's example [179]). The merit of graph-based models is that they are easily translated into object-oriented information model (as illustrated by the UML tool), this makes them fit well into the object-oriented programming environment prevalent nowadays. Also, since models developed using this approach can be easily mapped to

relational database, they offer efficient query processing and support for advanced query types.

On the other hand, the models developed relying on graphical notation have a lower degree of computer-evaluable formality, making them inadequate to provide powerful reasoning mechanisms. The graph-based models are to a large extent tools mainly used for human structuring purpose.

- ***Object-oriented models***

Common to object-oriented context modeling approaches is the intention to inherit main benefits of the object-oriented paradigm, namely encapsulation and reusability, to cover parts of the problems arising from the dynamicity of the context in ubiquitous computing environments. In such models, the details of context processing is encapsulated on an object level and therefore hidden to other components. Access to contextual information is provided through specified interfaces.

Object-oriented models are mainly used to support the abstraction of context information from sensors, rather than being a high-level model of context information that can be queried by context-aware applications. Representatives of the object-oriented model are the *cues* [180] developed within the TEA project [181] and the *active object model* of the GUIDE project [182]. Take *cues* for example, the concept of cues provides an abstraction from sensors. A cue is regarded as a function taking the value of a single sensor up to a certain time as input and producing output. The output of each cue depends on a single sensor, but different cues may be based on the same sensor. The context is modeled as an abstraction level on top of the available cues. Thus the cues in the programming model are objects providing contextual information through their interfaces, hiding the details of the process that produces output values.

Object-oriented models exhibit a certain degree of formality through the use of well-defined interfaces to access the object. The speed of information retrieval depends on the efficiency of underlying processing algorithms. Choosing object-oriented models has been primarily driven by the requirement of being able to manage a great variety of contextual information while maintaining scalability [182]. New types of contextual information (classes) as well as new or updated instances (objects) may be handled in the system in a distributed fashion. In addition, the applicability of this modeling approach to existing object-oriented ubiquitous computing runtime environments is given, but usually has strong additional requirements on resource-constrained computing devices — the requirements often cannot be fulfilled in ubiquitous computing systems.

- ***Logic-based model***

Logic-based modeling is a recent attempt to follow a formal way to represent context information. Logic-based models are distinct from the prior formal context models such as object-oriented and graphic-based models that build upon data modelling techniques developed by conventional information systems community. The logic-based context modelling approaches are primarily driven by the requirements of being able to manage a great variety of contextual information that is characterized by semantic-richness.

In logic-based models such as Sensed Context Model [183], the variety of contexts is addressed by way of rules and presuppositions (i.e. If *presuppositions* Then *rules*). Contextual information is modeled as expressions and rules. A set of rules defines the conditions on which a concluding expression may be derived (a process known as reasoning or inference) from a set of other expressions, or conditions that should be satisfied in order for a certain conclusion to be reached. A simple example of such rules is: the concluding expression "Bob is in meeting" is reached if the expression "Bob is now in room 206" and the expression "room 206 is holding a meeting" are true.

Logic-based models offer a high degree of formality, and are capable of describing contextual facts

and interrelationships in a precise and unambiguous manner, which allows powerful reasoning mechanisms to function. Indeed, the choice of adopting logic-based modeling approaches is driven primarily by employing their ability to assist context reasoning, rather than by context modeling itself (i.e. constructing a common data format to represent context information). Accordingly, logic-based modelling has been mainly used in conjunction with other modeling approaches as context reasoning mechanisms. For instance, Henricksen et al. [179] primarily takes on a graph-based context modelling approach, and uses a variant of predicate logic to derive higher-level contextual information.

A major weakness of this approach is its limited applicability to existing ubiquitous computing environments. Such models require the availability of logic reasoners and impose strong requirements on the resources of the computing devices, however full logic reasoners are usually not available on ubiquitous computing devices. Although logic-based approaches have an extremely high level of formality, validation is difficult due to the complexity of contextual interrelationships and difficulties to capture such intricacy. As a consequence, specification of contextual knowledge within a logic-based context model is very error-prone.

Appendix B: A introduction of the P3P and APPEL

This section introduces an important privacy practice, called "Platform for Privacy Preferences Project" (P3P) advocated by World Wide Web Consortium (W3C). The P3P is the first and influential "social protocol" to address privacy technically; Cranor and Reagle [184] call it a social protocol, because it mediates interactions between humans (through user agents and websites), in contrast to technical protocols, which facilitate machine-to-machine communication.

- ***The Platform for Privacy Preference Project (P3P)***

The Platform for Privacy Preferences Project (P3P) was launched in May 1997 at World Wide Web Consortium (W3C) in an effort to develop a specification and vocabulary to instruct websites to announce their privacy practices in a standardized machine-readable format that can be retrieved automatically and easily interpreted by users' browsers [95]. The user's web browser, by reading the privacy statement specified by the websites, can provide customized summaries to the user, or even take automated decisions (e.g., whether to block or allow placement of a certain cookie) on behalf of the user. In plain words, P3P tries to provide a means to communicate – in an electronic form – answers to questions such as "Who will get my personal data?", "Why is this data being collected?", or "How long will my information be stored?"

- ***P3P and APPEL***

A basic P3P architecture comprises user agents, privacy reference files, and privacy policies. When users access a website, their user agents obtain a privacy reference file for the website using one of several well-defined mechanisms. This file contains a list of mappings between URLs for the site's web resources and URLs of their associated

privacy policies. The web agent can thus ensure that the system downloads, parses, and compares the appropriate privacy policy with the user's preferences prior to accessing a web resource. P3P also specifies the language used to express privacy policies. While privacy preferences used to configure user agents can be expressed in several forms, P3P commonly uses APPEL [105], a similarly machine-readable preference language, to ensure that different user agents can reuse preferences. With the aid of APPEL, users can express personal preferences over most aspects of personal privacy and have automated processes to judge the acceptability of any such policy, or prompt for a decision instead. It is noticeable that P3P, however, does not attempt to enforce or ensure privacy through technology, for example, the cryptographic tools or anonymity techniques. Instead it relies on social and legal pressures to compel organizations to comply with their stated policies.

- ***Reasons of applying P3P to ubiquitous computing systems***

Recent years have seen increasing attempts to incorporate P3P into privacy protection framework for ubiquitous computing systems. It is also the author's interest to look into reasons behind, as well as to identify the viability and requirements of such a tool to be adopted in context-aware paradigm. There are basically three reasons:

First of all, P3P is the first complete and the most influential "social protocol" to address privacy technically. Significant work has gone into making P3P comply with existing and emerging legislation in information protection and privacy. Work built upon P3P could thus benefit from the substantial legal and social expertise that has been put into the development of this standard.

Secondly, although P3P is initially an attempt to find privacy mechanisms for the Web, its extension mechanism allows P3P policies to be adapted to suit ubiquitous computing environments. An important part of the P3P syntax is played by the EXTENSION element. It allows P3P to be arbitrarily extended, e.g., for adding application-specific

information or future backward-compatible extensions to the standard. Extensions can safely be ignored by user agents not familiar with the particular extension, unless the attribute optional="no" is given, in which case the extension is mandatory, and user agents not understanding it must ignore the whole policy.

Last but not least, under the umbrella of W3C, a number of auxiliary technologies have been constructed for P3P, most of which use platform-independent web technologies. Work building upon P3P can thus directly make use of related tools and libraries, and benefits from easy deployment and platform-independency.

Appendix C: A summary of the OECD Guidelines

1. Collection Limitation Principle. There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject.
2. Data Quality Principle. Personal data should be relevant to the purposes for which they are to be used, and, to the extent necessary for those purposes, should be accurate, complete and kept up-to-date.
3. Purpose Specification Principle. The purposes for which personal data are collected should be specified not later than at the time of data collection and the subsequent use limited to the fulfillment of those purposes or such others as are not incompatible with those purposes and as are specified on each occasion of change of purpose.
4. Use Limitation Principle. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with the Purpose Specification principle except:
   a) with the consent of the data subject; or
   b) by the authority of law.
5. Security Safeguards Principle. Personal data should be protected by reasonable security safeguards against such risks as loss or unauthorized access, destruction, use, modification or disclosure of data.
6. Openness Principle. There should be a general policy of openness about developments, practices and policies with respect to personal data. Means should be readily available of establishing the existence and nature of personal data, and the main purposes of their use, as well as the identity about usual residence of the data controller.
7. Individual Participation Principle. An individual should have the right:

a) to obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him;

b) to have communicated to him, data relating to him

   i. within a reasonable time;

   ii. at a charge, if any, that is not excessive;

   iii. in a reasonable manner; and

   iv. in a form that is readily intelligible to him;

c) to be given reasons if a request made under subparagraphs (a) and (b) is denied, and to be able to challenge such denial; and

d) to challenge data relating to him and, if the challenge is successful, to have the data erased; rectified, completed or amended.

8. Accountability Principle. A data controller should be accountable for complying with measures which give effect to the principles stated above.

Appendix D: A XML schema of the Privacy Policy/Preference Language

```
<?xml version='1.0' encoding='UTF-8'?>
<schema
  xmlns='http://www.w3.org/2001/XMLSchema'
  xmlns:pppl='http://ee.ucl.ac.uk/~jezhang/pppl'
  targetNamespace='http://ee.ucl.ac.uk/~jezhang/pppl'
  elementFormDefault='qualified'>

<!-- ********** POLICIES ********** -->
 <element name='POLICIES'>
  <complexType>
   <sequence>
    <element ref='pppl:POLICY' minOccurs='1' maxOccurs='unbounded'/>
   </sequence>
  </complexType>
 </element>

<!-- ********** POLICY ********** -->
<element name='POLICY'>
  <complexType>
   <sequence>
    <element ref='pppl:ENTITY' minOccurs='1' maxOccurs='1'/>
    <element ref='pppl:STATEMENT' minOccurs='1' maxOccurs='1'/>
   </sequence>
  </complexType>
 </element>

<!-- ********** ENTITY ********** -->
<element name='ENTITY'>
  <complexType>
   <sequence>
    <element name='pppl:APPLICATION' type="xsd:string" minOccurs='1' maxOccurs='1'/>
    <element name='pppl:ID' type="xsd:integer" minOccurs='1' maxOccurs='1'/>
    <element name='pppl:SERVICE-PROVIDER' type="xsd:string" minOccurs='1' maxOccurs='1'/>
    <element ref='pppl:CATEGORY' minOccurs='1' maxOccurs='1'/>
    <element ref='pppl:REQUEST-MODE' minOccurs='1' maxOccurs='1'/>
    <element ref='pppl:SERVICE-MODE' minOccurs='1' maxOccurs='1'/>
   </sequence>
  </complexType>
 </element>

<!-- ********** CATEGORY ********** -->
<element name="CATEGORY" type="pppl:CategoryTypeListRes" />
<simpleType name="CategoryTypeListRes">
   <restriction base="pppl:CategoryTypeList">
     <minLength value="1" />
     <minLength value="6" />
   </restriction>
</simpleType>
<simpleType name="CategoryTypeList">
  <list itemtype="pppl:CategoryType" />
</simpleType>
<simpleType  name="CategoryType">
     <restriction base="string">
       <enumeration value="tourism" />
       <enumeration value="finance" />
       <enumeration value="telecommunication" />
       <enumeration value="health" />
       <enumeration value="government" />
       <enumeration value="security" />
     </restriction>
</simpleType>
```

```
<!-- ************ REQUEST-MODE ************ -->
<element name="REQUEST-MODE">
  <simpleType>
    <restriction base="string">
      <enumeration value="continous" />
      <enumeration value="one-off" />
    </restriction>
  </simpleType>
</element>

<!-- ************ SERVICE-MODE ************ -->
<element name="SERVICE-MODE">
  <simpleType>
    <restriction base="string">
      <enumeration value="optional" />
      <enumeration value="mandatory" />
    </restriction>
  </simpleType>
</element>

<!-- ********** STATEMENT ********** -->
<element name='STATEMENT'>
  <complexType>
   <sequence>
    <element name='CONSEQUENCE' type='string' minOccurs='0' maxOccurs='1'/>
    <element ref='pppl:PURPOSE' minOccurs='1' maxOccurs='1'/>
    <element ref='pppl:RECIPIENT' minOccurs='1' maxOccurs='1'/>
    <element ref='pppl:RETENTION' minOccurs='1' maxOccurs='1'/>
    <element ref='pppl:DATA-GROUP' minOccurs='1' maxOccurs='1'/>
   </sequence>
  </complexType>
 </element>

<!-- ********** PURPOSE ********** -->
<element name="PURPOSE" type="pppl:PurposeTypeListRes " />
<simpleType name="PurposeTypeListRes ">
   <restriction base="PurposeTypeList ">
     <minLength value="1" />
     <maxLength value="9" />
   </restriction>
</simpleType>
<simpleType name="PurposeTypeList ">
  <list itemtype="pppl:PurposeType " />
</simpleType>
<simpleType  name="PurposeType" >
    <restriction base="string">
      <enumeration value="navigation" />
      <enumeration value="marketing" />
      <enumeration value="multimedia" />
      <enumeration value="communication" />
      <enumeration value="health" />
      <enumeration value="finance" />
      <enumeration value="social-analysis" />
      <enumeration value="develop" />
      <enumeration value="security" />
    </restriction>
  </simpleType>

<!-- ********** RECIPIENT ********** -->
<element name="RECIPIENT" type="pppl:RecipientTypeList "/>
<simpleType name="RecipientTypeList ">
  <list itemtype="pppl:RecipientType " />
</simpleType>
<simpleType  name="RecipientType ">
    <restriction base="string">
      <enumeration value="ours" />
      <enumeration value="same" />
      <enumeration value="other-recipient" />
      <enumeration value="delivery" />
      <enumeration value="public" />
      <enumeration value="unrelated" />
    </restriction>
```

```
</simpleType>

<!-- ********** RETENTION ********** -->
<element name="RETENTION" type="pppl:RetentionTypeList" />
<simpleType name="RetentionTypeList">
  <list itemtype="pppl:RetentionType" />
</simpleType>
<simpleType  name="RetentionType">
    <restriction base="string">
      <enumeration value="no-retention" />
      <enumeration value="stated-purpose" />
      <enumeration value="legal-requirement" />
      <enumeration value="business-practices" />
      <enumeration value="indefinitely"/>
    </restriction>
</simpleType>


<!-- ********** DATA-GROUP ********** -->
 <element name='DATA-GROUP' type='pppl:data-group-type' />
 <complexType name='data-group-type'>
  <any minOccurs='0' maxOccurs='unbounded'>
    <element name='DATA' type='pppl:data-in-statement ' maxOccurs='unbounded'/>
    <element name='TARGET-DATA' type='pppl:targetdata-in-statement ' maxOccurs='unbounded'/>
     <element name='OPTION' type='pppl:data-in-statement ' maxOccurs='unbounded'>
        <element name='DATA' type='pppl:data-in-statement ' minOccurs='1'
                 maxOccurs ='unbounded'/>
        <element name='TARGET-DATA' type='pppl:targetdata-in-statement ' minOccurs='1'
                 maxOccurs ='unbounded'/>
     </element>
  </any>
 </complexType>


</schema>
```

Appendix E: A list of the most common Jena inference rules used to detect preference conflict and redundancy[33]

```
# Conflict and Redundancy Detection Rule (Micro Version), context condition has at most
# two subsumption relations

@prefix pppl: <http://www.ee.ucl.ac.uk/~jezhang/privacypreferenceruleontology#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.

# @include <RDFS>.
# @include<OWL>.


# ------------  Rules to detect Modality Conflict (1) and Non Modality Conflict (11)-------

[ModalityConflict1: (?rulex pppl:modalityConflict ?ruley) <-
                    (?rulex rdf:type pppl:Rule)
                    (?ruley rdf:type pppl:Rule)
                    (?rulex pppl:hasBehavior ?bx)
                    (?ruley pppl:hasBehavior ?by)
                    notEqual(?bx ?by)
                    (?rulex pppl:hasData ?dx)
                    (?ruley pppl:hasData ?dy)
                    equal(?dx ?dy)
                    (?rulex pppl:hasPolicyCon ?pcx)
                    (?ruley pppl:hasPolicyCon ?pcy)
                    (?pcx pppl:equalTo_p ?pcy)
                    (?rulex pppl:hasContextCon ?ccx)
                    (?ruley pppl:hasContextCon ?ccy)
                    (?ccx pppl:equalTo_c ?ccy)
                    ]


[NonModalityConflict1: (?rulex pppl:nonModalityConflict ?ruley) <-
                       (?rulex rdf:type pppl:Rule)
                       (?ruley rdf:type pppl:Rule)
                       (?rulex pppl:hasBehavior ?bx)
                       (?ruley pppl:hasBehavior ?by)
                       notEqual(?bx ?by)
                       (?rulex pppl:hasData ?dx)
                       (?ruley pppl:hasData ?dy)
                       (?dx rdfs:subClassOf ?dy)
                       (?rulex pppl:hasPolicyCon ?pcx)
                       (?ruley pppl:hasPolicyCon ?pcy)
                       (?pcx pppl:equalTo_p ?pcy)
                       (?rulex pppl:hasContextCon ?ccx)
                       (?ruley pppl:hasContextCon ?ccy)
                       (?ccx pppl:equalTo_c ?ccy)
                      ]


[NonModalityConflict2: (?rulex pppl:nonModalityConflict ?ruley) <-
                       (?rulex rdf:type pppl:Rule)
                       (?ruley rdf:type pppl:Rule)
                       (?rulex pppl:hasBehavior ?bx)
                       (?ruley pppl:hasBehavior ?by)
                       notEqual(?bx ?by)
                       (?rulex pppl:hasData ?dx)
                       (?ruley pppl:hasData ?dy)
                       (?dy rdfs:subClassOf ?dx)
                       (?rulex pppl:hasPolicyCon ?pcx)
                       (?ruley pppl:hasPolicyCon ?pcy)
                       (?pcx pppl:equalTo_p ?pcy)
                       (?rulex pppl:hasContextCon ?ccx)
                       (?ruley pppl:hasContextCon ?ccy)
                       (?ccx pppl:equalTo_c ?ccy)
                      ]
```

[33] Section 6.4.1 describes briefly the development of these inference rules. The list of the inference rules is also available at http://ee.ucl.ac.uk/~jezhang/conflictdetectingrules_micro.rules

```
[NonModalityConflict3: (?rulex pppl:nonModalityConflict ?ruley) <-
                       (?rulex rdf:type pppl:Rule)
                       (?ruley rdf:type pppl:Rule)
                       (?rulex pppl:hasBehavior ?bx)
                       (?ruley pppl:hasBehavior ?by)
                       notEqual(?bx ?by)
                       (?rulex pppl:hasData ?dx)
                       (?ruley pppl:hasData ?dy)
                       equal(?dx ?dy)
                       (?rulex pppl:hasPolicyCon ?pcx)
                       (?ruley pppl:hasPolicyCon ?pcy)
                       (?pcx pppl:equalTo_p ?pcy)
                       (?rulex pppl:hasContextCon ?ccx)
                       (?ruley pppl:hasContextCon ?ccy)
                       (?ccx pppl:subsumedBy_c ?ccy)
                      ]

[NonModalityConflict4: (?rulex pppl:nonModalityConflict ?ruley) <-
                       (?rulex rdf:type pppl:Rule)
                       (?ruley rdf:type pppl:Rule)
                       (?rulex pppl:hasBehavior ?bx)
                       (?ruley pppl:hasBehavior ?by)
                       notEqual(?bx ?by)
                       (?rulex pppl:hasData ?dx)
                       (?ruley pppl:hasData ?dy)
                       equal(?dx ?dy)
                       (?rulex pppl:hasPolicyCon ?pcx)
                       (?ruley pppl:hasPolicyCon ?pcy)
                       (?pcx pppl:equalTo_p ?pcy)
                       (?rulex pppl:hasContextCon ?ccx)
                       (?ruley pppl:hasContextCon ?ccy)
                       (?ccy pppl:subsumedBy_c ?ccx)
                      ]

[NonModalityConflict5: (?rulex pppl:nonModalityConflict ?ruley) <-
                       (?rulex rdf:type pppl:Rule)
                       (?ruley rdf:type pppl:Rule)
                       (?rulex pppl:hasBehavior ?bx)
                       (?ruley pppl:hasBehavior ?by)
                       notEqual(?bx ?by)
                       (?rulex pppl:hasData ?dx)
                       (?ruley pppl:hasData ?dy)
                       (?dx rdfs:subClassOf ?dy)
                       (?rulex pppl:hasPolicyCon ?pcx)
                       (?ruley pppl:hasPolicyCon ?pcy)
                       (?pcx pppl:equalTo_p ?pcy)
                       (?rulex pppl:hasContextCon ?ccx)
                       (?ruley pppl:hasContextCon ?ccy)
                       (?ccx pppl:subsumedBy_c ?ccy)
                      ]


[testrule5: (?rulex pppl:testRule ?ruley) <-
                       (?rulex rdf:type pppl:Rule)
                       (?ruley rdf:type pppl:Rule)
                       (?rulex pppl:hasBehavior ?bx)
                       (?ruley pppl:hasBehavior ?by)
                       notEqual(?bx ?by)
                       (?rulex pppl:hasData ?dx)
                       (?ruley pppl:hasData ?dy)
                       (?dx rdfs:subClassOf ?dy)
                       (?rulex pppl:hasPolicyCon ?pcx)
                       (?ruley pppl:hasPolicyCon ?pcy)
                       (?pcx pppl:equalTo_p ?pcy)
                       (?rulex pppl:hasContextCon ?ccx)
                       (?ruley pppl:hasContextCon ?ccy)
                       (?ccx pppl:subsumedBy_c ?ccy)
                      ]


[NonModalityConflict6: (?rulex pppl:nonModalityConflict ?ruley) <-
                       (?rulex rdf:type pppl:Rule)
```

```
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        notEqual(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:subsumedBy_c ?ccx)
                       ]

[NonModalityConflict7:  (?rulex pppl:nonModalityConflict ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        notEqual(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccx pppl:subsumedBy_c ?ccy)
                       ]

[NonModalityConflict8:  (?rulex pppl:nonModalityConflict ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        notEqual(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dx rdfs:subClassOf ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:subsumedBy_c ?ccx)
                       ]

[NonModalityConflict9:  (?rulex pppl:nonModalityConflict ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        notEqual(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        equal(?dx ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccx pppl:mutuallySubsumedBy_c ?ccy)
                       ]

[NonModalityConflict10: (?rulex pppl:nonModalityConflict ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        notEqual(?bx ?by)
```

```
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                         (?dx rdfs:subClassOf ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccx pppl:mutuallySubsumedBy_c ?ccy)
                       ]


[NonModalityConflict11: (?rulex pppl:nonModalityConflict ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        notEqual(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                         (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccx pppl:mutuallySubsumedBy_c ?ccy)
                       ]


# ------------  Rules to detect redundancy of privacy preferences (6)------------------

[RedundancyDetecting1: (?rulex pppl:redundancyWith ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                         equal(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dx rdfs:subClassOf ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccx pppl:equalTo_c ?ccy)
                       ]


[RedundancyDetecting2: (?rulex pppl:redundancyWith ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccx pppl:equalTo_c ?ccy)
                       ]


[RedundancyDetecting3: (?rulex pppl:redundancyWith ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?bx ?by)
                        (?rulex pppl:hasData ?dx)
```

```
                        (?ruley pppl:hasData ?dy)
                        equal(?dx ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccx pppl:subsumedBy_c ?ccy)
                       ]


[RedundancyDetecting4: (?rulex pppl:redundancyWith ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        equal(?dx ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:subsumedBy_c ?ccx)
                       ]


[RedundancyDetecting5: (?rulex pppl:redundancyWith ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dx rdfs:subClassOf ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccx pppl:subsumedBy_c ?ccy)
                       ]


[RedundancyDetecting6: (?rulex pppl:redundancyWith ?ruley) <-
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?rulex pppl:hasBehavior ?bx)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?bx ?by)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:equalTo_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:subsumedBy_c ?ccx)
                       ]



# ------------  The definition of equalTo_p (8 situations, complete) --------

# 1. both rules don't have any policy condition

[equalPolicyCondition1:  (?pcx pppl:equalTo_p ?pcy)  <-
                         (?pcx rdf:type pppl:PolicyCondition)
                         (?pcy rdf:type pppl:PolicyCondition)
                         noValue(?pcx pppl:hasPurpose ?purx)
                         noValue(?pcx pppl:hasRecipient ?recx)
```

```
                        noValue(?pcx pppl:hasRetention ?retx)
                        noValue(?pcy pppl:hasPurpose ?pury)
                        noValue(?pcy pppl:hasRecipient ?recy)
                        noValue(?pcy pppl:hasRetention ?rety)
                       ]


# 2. both rules only have Purpose Condition, and with equal value

[equalPolicyCondition2:  (?pcx pppl:equalTo_p ?pcy)  <-
                        (?pcx rdf:type pppl:PolicyCondition)
                        (?pcy rdf:type pppl:PolicyCondition)
                        (?pcx pppl:hasPurpose ?purx)
                        noValue(?pcx pppl:hasRecipient ?recx)
                        noValue(?pcx pppl:hasRetention ?retx)
                        (?pcy pppl:hasPurpose ?pury)
                        noValue(?pcy pppl:hasRecipient ?recy)
                        noValue(?pcy pppl:hasRetention ?rety)
                        equal(?purx ?pury)
                        ]



# 3. both rules only has Recipient Condition, and with equal value

[equalPolicyCondition3:  (?pcx pppl:equalTo_p ?pcy)  <-
                        (?pcx rdf:type pppl:PolicyCondition)
                        (?pcy rdf:type pppl:PolicyCondition)
                         noValue(?pcx pppl:hasPurpose ?purx)
                        (?pcx pppl:hasRecipient ?recx)
                        noValue(?pcx pppl:hasRetention ?retx)
                        noValue(?pcy pppl:hasPurpose ?pury)
                        (?pcy pppl:hasRecipient ?recy)
                        noValue(?pcy pppl:hasRetention ?rety)
                        equal(?recx ?recy)
                        ]


# 4. both rules only have Retention Condition, and with equal value.

[equalPolicyCondition4:  (?pcx pppl:equalTo_p ?pcy)  <-
                        (?pcx rdf:type pppl:PolicyCondition)
                        (?pcy rdf:type pppl:PolicyCondition)
                        noValue(?pcx pppl:hasPurpose ?purx)
                        noValue(?pcx pppl:hasRecipient ?recx)
                        (?pcx pppl:hasRetention ?retx)
                        noValue(?pcy pppl:hasPurpose ?pury)
                        noValue(?pcy pppl:hasRecipient ?recy)
                        (?pcy pppl:hasRetention ?rety)
                        equal(?retx ?rety)
                        ]


# 5. both rules only have Purpose and Recipient Conditions, and with equal values

[equalPolicyCondition5:  (?pcx pppl:equalTo_p ?pcy)  <-
                        (?pcx rdf:type pppl:PolicyCondition)
                        (?pcy rdf:type pppl:PolicyCondition)
                        (?pcx pppl:hasPurpose ?purx)
                        (?pcx pppl:hasRecipient ?recx)
                        noValue(?pcx pppl:hasRetention ?retx)
                        (?pcy pppl:hasPurpose ?pury)
                        (?pcy pppl:hasRecipient ?recy)
                        noValue(?pcy pppl:hasRetention ?rety)
                        equal(?purx ?pury)
                        equal(?recx ?recy)
                        ]



# 6. both rules only have Purpose and Retention Conditions, and with equal values

[equalPolicyCondition6:  (?pcx pppl:equalTo_p ?pcy)  <-
                        (?pcx rdf:type pppl:PolicyCondition)
                        (?pcy rdf:type pppl:PolicyCondition)
                        (?pcx pppl:hasPurpose ?purx)
```

```
                        noValue(?pcx pppl:hasRecipient ?recx)
                        (?pcx pppl:hasRetention ?retx)
                        (?pcy pppl:hasPurpose ?pury)
                        noValue(?pcy pppl:hasRecipient ?recy)
                        (?pcy pppl:hasRetention ?rety)
                        equal(?purx ?pury)
                        equal(?retx ?rety)
                       ]


# 7. both rules only have Recipient and Retention Conditions, and with equal values

[equalPolicyCondition7:  (?pcx pppl:equalTo_p ?pcy)  <-
                        (?pcx rdf:type pppl:PolicyCondition)
                        (?pcy rdf:type pppl:PolicyCondition)
                        noValue(?pcx pppl:hasPurpose ?purx)
                        (?pcx pppl:hasRecipient ?recx)
                        (?pcx pppl:hasRetention ?retx)
                        noValue(?pcy pppl:hasPurpose ?pury)
                        (?pcy pppl:hasRecipient ?recy)
                        (?pcy pppl:hasRetention ?rety)
                        equal(?recx ?recy)
                        equal(?retx ?rety)
                        ]


# 8. both rules have all three policy conditions and with equal value

[equalPolicyCondition8:  (?pcx pppl:equalTo_p ?pcy)  <-
                        (?pcx rdf:type pppl:PolicyCondition)
                        (?pcy rdf:type pppl:PolicyCondition)
                        (?pcx pppl:hasPurpose ?purx)
                        (?pcx pppl:hasRecipient ?recx)
                        (?pcx pppl:hasRetention ?retx)
                        (?pcy pppl:hasPurpose ?pury)
                        (?pcy pppl:hasRecipient ?recy)
                        (?pcy pppl:hasRetention ?rety)
                        equal(?purx ?pury)
                        equal(?recx ?recy)
                        equal(?retx ?rety)
                        ]


# ------------  The definition of equalTo_c (16) ------------------

# 1. both rules don't have any context conditions

[equalContextCondition1: (?ccx pppl:equalTo_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                       ]


# 2. both rules only have Target Condition, and with equal value

[equalContextCondition2: (?ccx pppl:equalTo_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
```

```
                        equal(?tax ?tay)
                        ]


# 3. both rules only have Location Condition, and with equal value

[equalContextCondition3: (?ccx pppl:equalTo_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?lox ?loy)
                        ]


# 4. both rules only have Time Condition, and with equal value

[equalContextCondition4: (?ccx pppl:equalTo_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?timx ?timy)
                        ]


# 5. both rules only have Activity Condition, and with equal value

[equalContextCondition5: (?ccx pppl:equalTo_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?actx ?acty)
                        ]


# 6. both rules only have Target and Location Conditions, and with equal values

[equalContextCondition6: (?ccx pppl:equalTo_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?tax ?tay)
                        equal(?lox ?loy)
                        ]


# 7. both rules only have Target and Time Conditions, and with equal values

[equalContextCondition7: (?ccx pppl:equalTo_c ?ccy) <-
```

```
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?timx ?timy)
                          ]


# 8. both rules only have Target and Activity Conditions, and with equal values

[equalContextCondition8: (?ccx pppl:equalTo_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?actx ?acty)
                          ]


# 9. both rules only have Location and Time Conditions, and with equal values

[equalContextCondition9: (?ccx pppl:equalTo_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?lox ?loy)
                          equal(?timx ?timy)
                          ]


# 10. both rules only have Location and Activity Conditions, and with equal values

[equalContextCondition10:(?ccx pppl:equalTo_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          equal(?lox ?loy)
                          equal(?actx ?acty)
                          ]


# 11. both rules only have Time and Activity Conditions, and with equal values

[equalContextCondition11:(?ccx pppl:equalTo_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
```

```
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          equal(?timx ?timy)
                          equal(?actx ?acty)
                          ]


# 12. both rules only have Target, Location, and Time Conditions, and with equal values

[equalContextCondition12: (?ccx pppl:equalTo_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?lox ?loy)
                          equal(?timx ?timy)
                          ]

# 13. both rules only have Target, Location, and Activity Conditions, and with equal value

[equalContextCondition13: (?ccx pppl:equalTo_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?lox ?loy)
                          equal(?actx ?acty)
                          ]


# 14. both rules only have Target, Time, and Activity Conditions, and with equal values

[equalContextCondition14: (?ccx pppl:equalTo_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?timx ?timy)
                          equal(?actx ?acty)
                          ]


# 15. both rules only have Location, Time, and Activity Conditions, and with equal values

[equalContextCondition15: (?ccx pppl:equalTo_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
```

```
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?lox ?loy)
                        equal(?timx ?timy)
                        equal(?actx ?acty)
                        ]


# 16. both rules have all context conditions, and with equal value

[equalContextCondition16: (?ccx pppl:equalTo_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?tax ?tay)
                        equal(?lox ?loy)
                        equal(?timx ?timy)
                        equal(?actx ?acty)
                        ]



# ------------  The definition of subsume_c & subsumedBy_c (124 ) -------------------
# Micro Version: 4*2 + 6*4 + 15*4 + 32 = 92 + 32= 124,
# Full Version: Mirco + 1*4 + 1*5= 124 + 4 +5 = 133,
# Mini version 2*4+3*4=20 (subset of Mirco, and if having two dimensions, must have target .


# 1.a both rules only have Target Condition, and with value subsumed.

[subsumeContextCondition1a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                       ]


# 1.b one rule only have Target Condition, and another rule don't have any context conditio

[subsumeContextCondition1b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                       ]



# 2.a both rules only have Location Condition, and with value subsumed.

[subsumeContextCondition2a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
```

```
                          noValue(?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?lox rdfs:subClassOf ?loy)
                         ]


# 2.b one rule only has Location Condition, and another rule don't have any context conditi

[subsumeContextCondition2b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          ]


# 3.a both rules only has Time Condition, and with value subsumed.

[subsumeContextCondition3a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?timx rdfs:subClassOf ?timy)
                          ]

# 3.b one rule only has Time Condition, and another rule don't have any context condition.

[subsumeContextCondition3b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          ]


# 4.a both rules only has Activity Condition, and with value subsumed.

[subsumeContextCondition4a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          ]
```

```
# 4.b one rule only has Activity Condition, and another rule don't have any context conditi

[subsumeContextCondition4b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          ]


# 5.a both rules only has Target and Location Conditions, and with value subsumed.

[subsumeContextCondition5a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          (?lox rdfs:subClassOf ?loy)
                          ]


# 5.b one rule only has Target and Location Conditions, and another rule only has Location
#     condition with value subsuming the first rule.

[subsumeContextCondition5b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?lox rdfs:subClassOf ?loy)
                          ]


# 5.c one rule only has Target and Location Conditions, and another rule only has Target
#     condition with value subsuming the first rule.

[subsumeContextCondition5c: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          ]

# 5.d one rule only has Target and Location Conditions, and another rule don't have any
#     context condition.

[subsumeContextCondition5d: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
```

```
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        ]


# 6.a both rules only have Target and Time Conditions, and with value subsumed.

[subsumeContextCondition6a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?timx rdfs:subClassOf ?timy)
                        ]


#6.b  one rule only has Target and Time Conditions, and another rule only has Time conditic
#     with value subsuming the first rule.

[subsumeContextCondition6b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        ]


#6.c one rule only has Target and Time Conditions, and another rule only has Target conditi
#   with value subsuming the first rule.

[subsumeContextCondition6c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        ]


#6.d one rule only has Target and Time Conditions, and another rule don't have any context
# condition.

[subsumeContextCondition6d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
```

```
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        ]


# 7.a both rules only have Target and Activity Conditions, and with value subsumed.

[subsumeContextCondition7a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?actx rdfs:subClassOf ?acty)
                        ]


#7.b  one rule only has Target and Activity Conditions, and another rule only has Activity
#     condition with value subsuming the first rule.

[subsumeContextCondition7b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        ]


#7.c one rule only has Target and Activity Conditions, and another rule only has Target
#    condition with value subsuming the first rule.

[subsumeContextCondition7c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        ]


#7.d one rule only has Target and Activity Conditions, and another rule don't have any
#    context condition.

[subsumeContextCondition7d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                       ]
```

```
# 8.a both rules only have Location and Time Conditions, and with value subsumed.

[subsumeContextCondition8a: (?ccx pppl:subsumedBy_c ?ccy) <-
                            (?ccx rdf:type pppl:ContextCondition)
                            (?ccy rdf:type pppl:ContextCondition)
                            noValue(?ccx pppl:hasTarget ?tax)
                            (?ccx pppl:hasLocation ?lox)
                            (?ccx pppl:hasTime ?timx)
                            noValue(?ccx pppl:hasActivity ?actx)
                            noValue(?ccy pppl:hasTarget ?tay)
                            (?ccy pppl:hasLocation ?loy)
                            (?ccy pppl:hasTime ?timy)
                            noValue(?ccy pppl:hasActivity ?acty)
                            (?lox rdfs:subClassOf ?loy)
                            (?timx rdfs:subClassOf ?timy)
                            ]


# 8.b one rule only has Location and Time Conditions, and another rule only has Time
#     condition with value subsuming the first rule.

[subsumeContextCondition8b: (?ccx pppl:subsumedBy_c ?ccy) <-
                            (?ccx rdf:type pppl:ContextCondition)
                            (?ccy rdf:type pppl:ContextCondition)
                            noValue(?ccx pppl:hasTarget ?tax)
                            (?ccx pppl:hasLocation ?lox)
                            (?ccx pppl:hasTime ?timx)
                            noValue(?ccx pppl:hasActivity ?actx)
                            noValue(?ccy pppl:hasTarget ?tay)
                            noValue(?ccy pppl:hasLocation ?loy)
                            (?ccy pppl:hasTime ?timy)
                            noValue(?ccy pppl:hasActivity ?acty)
                            (?timx rdfs:subClassOf ?timy)
                            ]


# 8.c one rule only has Location and Time  Conditions, and another rule only has Location
#     condition with value subsuming the first rule.

[subsumeContextCondition8c: (?ccx pppl:subsumedBy_c ?ccy) <-
                           (?ccx rdf:type pppl:ContextCondition)
                           (?ccy rdf:type pppl:ContextCondition)
                           noValue(?ccx pppl:hasTarget ?tax)
                           (?ccx pppl:hasLocation ?lox)
                           (?ccx pppl:hasTime ?timx)
                           noValue(?ccx pppl:hasActivity ?actx)
                           noValue(?ccy pppl:hasTarget ?tay)
                           (?ccy pppl:hasLocation ?loy)
                           noValue(?ccy pppl:hasTime ?timy)
                           noValue(?ccy pppl:hasActivity ?acty)
                           (?lox rdfs:subClassOf ?loy)
                           ]


# 8.d one rule only has Location and Time  Conditions, and another rule don't have any
#     context condition.

[subsumeContextCondition8d: (?ccx pppl:subsumedBy_c ?ccy) <-
                           (?ccx rdf:type pppl:ContextCondition)
                           (?ccy rdf:type pppl:ContextCondition)
                           noValue(?ccx pppl:hasTarget ?tax)
                           (?ccx pppl:hasLocation ?lox)
                           (?ccx pppl:hasTime ?timx)
                           noValue(?ccx pppl:hasActivity ?actx)
                           noValue(?ccy pppl:hasTarget ?tay)
                           noValue(?ccy pppl:hasLocation ?loy)
                           noValue(?ccy pppl:hasTime ?timy)
                           noValue(?ccy pppl:hasActivity ?acty)
                           ]


# 9.a both rules only have Location and Activity Conditions, and with value subsumed.
```

```
[subsumeContextCondition9a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        (?actx rdfs:subClassOf ?acty)
                        ]


# 9.b  one rule only has Location and Activity Conditions, and another rule only has
#      Activity condition with value subsuming the first rule.

[subsumeContextCondition9b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        ]


# 9.c  one rule only has Location and Activity Conditions, and another rule only has
#      Location condition with value subsuming the first rule.

[subsumeContextCondition9c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                       ]


# 9.d  one rule only has Location and Activity Conditions, and another rule don't have any
#      context condition.

[subsumeContextCondition9d: (?ccx pppl:subsumedBy_c ?ccy) <-
                       (?ccx rdf:type pppl:ContextCondition)
                       (?ccy rdf:type pppl:ContextCondition)
                       noValue(?ccx pppl:hasTarget ?tax)
                       (?ccx pppl:hasLocation ?lox)
                       noValue(?ccx pppl:hasTime ?timx)
                       (?ccx pppl:hasActivity ?actx)
                       noValue(?ccy pppl:hasTarget ?tay)
                       noValue(?ccy pppl:hasLocation ?loy)
                       noValue(?ccy pppl:hasTime ?timy)
                       noValue(?ccy pppl:hasActivity ?acty)
                      ]


# 10.a both rules only have Time and Activity Conditions, and with value subsumed.

[subsumeContextCondition10a: (?ccx pppl:subsumedBy_c ?ccy) <-
                       (?ccx rdf:type pppl:ContextCondition)
                       (?ccy rdf:type pppl:ContextCondition)
```

```
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        (?actx rdfs:subClassOf ?acty)
                        ]


# 10.b  one rule only has Time and Activity Conditions, and another rule only has Activity
#       condition with value subsuming the first rule.

[subsumeContextCondition10b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        ]


# 10.c  one rule only has Time and Activity Conditions, and another rule only has Time
#       condition with value subsuming the first rule.

[subsumeContextCondition10c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        ]


# 10.d  one rule only has Time and Activity Conditions, and another rule don't have any
#       context condition.

[subsumeContextCondition10d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        ]


# 11.a both rules only have Target, Location, and Time Conditions, and with value subsumed.
# this is not included in micro version


# 11.b one rule only has Target, Location, and Time Conditions, two of these conditions are
#      with value subsumed while another condition has equal value


# 11.b.1.a  Target and Location conditions of two rules are with value subsumed, while Time
#           condition of two rules has equal value.
```

```
[subsumeContextCondition_11b1a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?lox rdfs:subClassOf ?loy)
                        equal(?timx ?timy)
                        ]


# 11.b.1.b one rule only has Target and Location Conditions, and another rule only has Targ
#   condition with value subsuming the first rule, while Time condition of two rules are eq

[subsumeContextCondition_11b1b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        equal(?timx ?timy)
                        ]


# 11.b.1.c one rule only has Target and Location Conditions, and another rule only has Loca
#   condition with value subsuming the first rule, while Time condition of two rules are eq

[subsumeContextCondition_11b1c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        equal(?timx ?timy)
                        ]


# 11.b.1.d one rule only has Target and Location Conditions, and another rule don't have  T
#          and Location conditions, while Time condition of two rules are equal.

[subsumeContextCondition_11b1d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?timx ?timy)
                        ]


# 11.b.2.a  Target and Time conditions of two rules are with value subsumed, while Location
#           condition of two rules has equal value.
```

```
[subsumeContextCondition_11b2a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?lox ?loy)
                        ]


# 11.b.2.b one rule only has Target and Time Conditions, and another rule only has Target c
#          with value subsuming the first rule, while Location condition of two rules are e

[subsumeContextCondition_11b2b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        equal(?lox ?loy)
                        ]


# 11.b.2.c one rule only has Target and Time Conditions, and another rule only has Time con
#          with value subsuming the first rule, while Location condition of two rules are e

[subsumeContextCondition_11b2c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?lox ?loy)
                        ]


# 11.b.2.d one rule only has Target and Time Conditions, and another rule don't have Target
#          and Time conditions, while Location condition of two rules are equal.

[subsumeContextCondition_11b2d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?lox ?loy)
                        ]


# 11.b.3.a  Location and Time conditions of two rules are with value subsumed, while Target
```

```
#          condition of two rules has equal value.

[subsumeContextCondition_11b3a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?lox rdfs:subClassOf ?loy)
                          (?timx rdfs:subClassOf ?timy)
                          equal(?tax ?tay)
                          ]


# 11.b.3.b one rule only has Location and Time Conditions, and another rule only has Locati
# condition with value subsuming the first rule, while Target condition of two rules are eq

[subsumeContextCondition_11b3b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?lox rdfs:subClassOf ?loy)
                          equal(?tax ?tay)
                          ]


# 11.b.3.c one rule only has Location and Time Conditions, and another rule only has Time
# condition with value subsuming the first rule, while Target condition of two rules are eq

[subsumeContextCondition_11b3c: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?timx rdfs:subClassOf ?timy)
                          equal(?tax ?tay)
                          ]


# 11.b.3.d one rule only has Location and Time Conditions, and another rule don't have Time
#          and Location condition, while Target condition of two rules are equal.

[subsumeContextCondition_11b3d: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          ]
```

```
# 11.c one rule only has Target, Location, and Time Conditions, one of these conditions is
#      with value subsumed while other two conditions have equal value

# 11.c.1  Target condition of two rules is with value subsumed, while Location and Time
#         conditions of two rules have equal value.

[subsumeContextCondition_11c1: (?ccx pppl:subsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         (?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         noValue(?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         (?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         noValue(?ccy pppl:hasActivity ?acty)
                         (?tax rdfs:subClassOf ?tay)
                         equal(?lox ?loy)
                         equal(?timx ?timy)
                         ]

# 11.c.2  Location condition of two rules is with value subsumed, while Target and Time
#         conditions of two rules have equal value.

[subsumeContextCondition_11c2: (?ccx pppl:subsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         (?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         noValue(?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         (?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         noValue(?ccy pppl:hasActivity ?acty)
                         (?lox rdfs:subClassOf ?loy)
                         equal(?tax ?tay)
                         equal(?timx ?timy)
                         ]

# 11.c.3  Time condition of two rules is with value subsumed, while Target and Location
#         conditions of two rules have equal value.

[subsumeContextCondition3: (?ccx pppl:subsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         (?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         noValue(?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         (?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         noValue(?ccy pppl:hasActivity ?acty)
                         (?timx rdfs:subClassOf ?timy)
                         equal(?tax ?tay)
                         equal(?lox ?loy)
                         ]


# 12.a both rules only have Target, Location, and Activity Conditions, and with value subsu
# this is not in micro version


# 12.b one rule only has Target, Location, and Activity Conditions, two of these conditions
#      are with value subsumed while another condition has equal value

# 12.b.1.a  Target and Location conditions of two rules are with value subsumed, while
#           Activity condition of two rules has equal value.

[subsumeContextCondition_12b1a: (?ccx pppl:subsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
```

```
                            (?ccx pppl:hasLocation ?lox)
                            noValue(?ccx pppl:hasTime ?timx)
                            (?ccx pppl:hasActivity ?actx)
                            (?ccy pppl:hasTarget ?tay)
                            (?ccy pppl:hasLocation ?loy)
                            noValue(?ccy pppl:hasTime ?timy)
                            (?ccy pppl:hasActivity ?acty)
                            (?tax rdfs:subClassOf ?tay)
                            (?lox rdfs:subClassOf ?loy)
                            equal(?actx ?acty)
                            ]


# 12.b.1.b one rule only has Target and Location Conditions, and another rule only has Targ
# condition with value subsuming the first rule, while Activity condition of two rules are

[subsumeContextCondition_12b1b: (?ccx pppl:subsumedBy_c ?ccy) <-
                            (?ccx rdf:type pppl:ContextCondition)
                            (?ccy rdf:type pppl:ContextCondition)
                            (?ccx pppl:hasTarget ?tax)
                            (?ccx pppl:hasLocation ?lox)
                            noValue(?ccx pppl:hasTime ?timx)
                            (?ccx pppl:hasActivity ?actx)
                            (?ccy pppl:hasTarget ?tay)
                            noValue(?ccy pppl:hasLocation ?loy)
                            noValue(?ccy pppl:hasTime ?timy)
                            (?ccy pppl:hasActivity ?acty)
                            (?tax rdfs:subClassOf ?tay)
                            equal(?actx ?acty)
                            ]

# 12.b.1.c one rule only has Target and Location Conditions, and another rule only has Loca
# condition with value subsuming the first rule, while Activity condition of two rules are

[subsumeContextCondition_12b1c: (?ccx pppl:subsumedBy_c ?ccy) <-
                            (?ccx rdf:type pppl:ContextCondition)
                            (?ccy rdf:type pppl:ContextCondition)
                            (?ccx pppl:hasTarget ?tax)
                            (?ccx pppl:hasLocation ?lox)
                            noValue(?ccx pppl:hasTime ?timx)
                            (?ccx pppl:hasActivity ?actx)
                            noValue(?ccy pppl:hasTarget ?tay)
                            (?ccy pppl:hasLocation ?loy)
                            noValue(?ccy pppl:hasTime ?timy)
                            (?ccy pppl:hasActivity ?acty)
                            (?lox rdfs:subClassOf ?loy)
                            equal(?actx ?acty)
                            ]


# 12.b.1.d one rule only has Target and Location Conditions, and another rule don't have
#          Target and Location conditions, while Activity condition of two rules are equal.

[subsumeContextCondition_12b1d: (?ccx pppl:subsumedBy_c ?ccy) <-
                            (?ccx rdf:type pppl:ContextCondition)
                            (?ccy rdf:type pppl:ContextCondition)
                            (?ccx pppl:hasTarget ?tax)
                            (?ccx pppl:hasLocation ?lox)
                            noValue(?ccx pppl:hasTime ?timx)
                            (?ccx pppl:hasActivity ?actx)
                            noValue(?ccy pppl:hasTarget ?tay)
                            noValue(?ccy pppl:hasLocation ?loy)
                            noValue(?ccy pppl:hasTime ?timy)
                            (?ccy pppl:hasActivity ?acty)
                            equal(?actx ?acty)
                            ]


# 12.b.2.a  Target and Activity conditions of two rules are with value subsumed, while
#           Location condition of two rules has equal value.

[subsumeContextCondition_12b2a: (?ccx pppl:subsumedBy_c ?ccy) <-
                            (?ccx rdf:type pppl:ContextCondition)
                            (?ccy rdf:type pppl:ContextCondition)
                            (?ccx pppl:hasTarget ?tax)
                            (?ccx pppl:hasLocation ?lox)
```

```
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          (?actx rdfs:subClassOf ?acty)
                          equal(?lox ?loy)
                          ]


# 12.b.2.b one rule only has Target and Activity Conditions, and another rule only has Targ
#condition with value subsuming the first rule, while Location condition of two rules are e

[subsumeContextCondition_12b2b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          equal(?lox ?loy)
                          ]


# 12.b.2.c one rule only has Target and Activity Conditions, and another rule only has
#          Activity condition with value subsuming the first rule, while Location condition
#           of two rules are equal.

[subsumeContextCondition_12b2c: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?actx rdfs:subClassOf ?acty)
                          equal(?lox ?loy)
                          ]


# 12.b.2.d one rule only has Target and Activity Conditions, and another rule don't have
#          Target and Activity conditions, while Location condition of two rules are equal.

[subsumeContextCondition_12b2d: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?lox ?loy)
                          ]


# 12.b.3.a  Location and Activity conditions of two rules are with value subsumed, while
#           Target condition of two rules has equal value.

[subsumeContextCondition_12b3a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
```

```
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?tax ?tay)
                        ]


# 12.b.3.b one rule only has Location and Activity Conditions, and another rule only has
#          Location condition with value subsuming the first rule, while Target condition
#          of two rules are equal.

[subsumeContextCondition_12b3b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        equal(?tax ?tay)
                        ]


# 12.b.3.c one rule only has Location and Activity Conditions, and another rule only has
#          Activity condition with value subsuming the first rule, while Target condition
#          of two rules are equal.

[subsumeContextCondition_12b3c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?tax ?tay)
                        ]


# 12.b.3.d one rule only has Location and Activity Conditions, and another rule don't have
#          Activity and Location condition, while Target condition of two rules are equal.

[subsumeContextCondition_12b3d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?tax ?tay)
                        ]


# 12.c one rule only has Target, Location, and Activity Conditions, one of these conditions
#      is with value subsumed while other two conditions have equal value
```

```
# 12.c.1  Target condition of two rules is with value subsumed, while Location and Activity
#         conditions of two rules have equal value.

[subsumeContextCondition_12c1: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        equal(?lox ?loy)
                        equal(?actx ?acty)
                        ]


# 12.c.2  Location condition of two rules is with value subsumed, while Target and Activity
#         conditions of two rules have equal value.

[subsumeContextCondition_12c2: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?loc rdfs:subClassOf ?loy)
                        equal(?tax ?tay)
                        equal(?actx ?acty)
                        ]


# 12.c.3  Activity condition of two rules is with value subsumed, while Target and Location
#         conditions of two rules have equal value.

[subsumeContextCondition_12c3: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?tax ?tay)
                        equal(?lox ?loy)
                        ]



# 13.a both rules only have Target, Time, and Activity Conditions, and with value subsumed.
# this is not in micro version


# 13.b one rule only has Target, Time, and Activity Conditions, two of these conditions are
#      with value subsumed while another condition has equal value


# 13.b.1.a  Target and Time conditions of two rules are with value subsumed, while Activity
#           condition of two rules has equal value.

[subsumeContextCondition_13b1a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
```

```
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?act ?acty)
                        ]


# 13.b.1.b one rule only has Target and Time Conditions, and another rule only has Target
#          condition with value subsuming the first rule, while Activity condition of two
#          rules are equal.

[subsumeContextCondition_13b1b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        equal(?act ?acty)
                        ]


# 13.b.1.c one rule only has Target and Time Conditions, and another rule only has Location
#          condition with value subsuming the first rule, while Activity condition of two
#          rules are equal.

[subsumeContextCondition_13b1c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?act ?acty)
                        ]


# 13.b.1.d one rule only has Target and Time Conditions, and another rule don't have Target
#          and Time conditions, while Activity condition of two rules are equal.

[subsumeContextCondition_13b1d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?act ?acty)
                        ]


# 13.b.2.a  Target and Activity conditions of two rules are with value subsumed, while Time
#           condition of two rules has equal value.
```

```
[subsumeContextCondition_13b2a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?timx ?timy)
                        ]


# 13.b.2.b one rule only has Target and Activity Conditions, and another rule only has
#          Target condition with value subsuming the first rule, while Time condition
#          of two rules are equal.

[subsumeContextCondition_13b2b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        equal(?timx ?timy)
                        ]


# 13.b.2.c one rule only has Target and Activity Conditions, and another rule only has
#          Activity condition with value subsuming the first rule, while Time condition
#          of two rules are equal.

[subsumeContextCondition_13b2c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?timx ?timy)
                        ]


# 13.b.2.d one rule only has Target and Activity Conditions, and another rule don't have
#          Target and Activity conditions, while Time condition of two rules are equal.

[subsumeContextCondition_13b2d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?timx ?timy)
                        ]


# 13.b.3.a  Time and Activity conditions of two rules are with value subsumed, while Target
```

```
#           condition of two rules has equal value.

[subsumeContextCondition_13b3a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?timx rdfs:subClassOf ?timy)
                          (?actx rdfs:subClassOf ?acty)
                          equal(?tax ?tay)
                          ]


# 13.b.3.b one rule only has Time and Activity Conditions, and another rule only has
#          Time condition with value subsuming the first rule, while Target condition
#          of two rules are equal.

[subsumeContextCondition_13b3b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?timx rdfs:subClassOf ?timy)
                          equal(?tax ?tay)
                          ]


# 13.b.3.c one rule only has Time and Activity Conditions, and another rule only has
#          Activity condition with value subsuming the first rule, while Target condition
#          of two rules are equal.

[subsumeContextCondition_13b3c: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?actx rdfs:subClassOf ?acty)
                          equal(?tax ?tay)
                          ]


# 13.b.3.d one rule only has Time and Activity Conditions, and another rule don't have
#          Activity and Time condition, while Target condition of two rules are equal.

[subsumeContextCondition_13b3d: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          ]
```

```
# 13.c one rule only has Target, Time, and Activity Conditions, one of these conditions is
#      with value subsumed while other two conditions have equal value


# 13.c.1  Target condition of two rules is with value subsumed, while Time and Activity
#         conditions of two rules have equal value.

[subsumeContextCondition_13c1: (?ccx pppl:subsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         noValue(?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         (?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         noValue(?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         (?ccy pppl:hasActivity ?acty)
                         (?tax rdfs:subClassOf ?tay)
                         equal(?timx ?timy)
                         equal(?actx ?acty)
                         ]


# 13.c.2  Time condition of two rules is with value subsumed, while Target and Activity
#         conditions of two rules have equal value.

[subsumeContextCondition_13c2: (?ccx pppl:subsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         noValue(?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         (?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         noValue(?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         (?ccy pppl:hasActivity ?acty)
                         (?timx rdfs:subClassOf ?timy)
                         equal(?tax ?tay)
                         equal(?actx ?acty)
                         ]


# 13.c.3  Activity condition of two rules is with value subsumed, while Target and Time
#         conditions of two rules have equal value.

[subsumeContextCondition_13c3: (?ccx pppl:subsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         noValue(?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         (?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         noValue(?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         (?ccy pppl:hasActivity ?acty)
                         (?actx rdfs:subClassOf ?acty)
                         equal(?tax ?tay)
                         equal(?timx ?timy)
                         ]


# 14.a both rules only have Location, Time, and Activity Conditions, and with value subsume
# this is not in micro version


# 14.b one rule only has Location, Time, and Activity Conditions, two of these conditions
#      are with value subsumed while another condition has equal value

# 14.b.1.a  Location and Time conditions of two rules are with value subsumed, while
#           Activity  condition of two rules has equal value.

[subsumeContextCondition_14b1a: (?ccx pppl:subsumedBy_c ?ccy) <-
```

```
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?actx ?acty)
                        ]


# 14.b.1.b one rule only has Location and Time Conditions, and another rule only has
#          Location condition with value subsuming the first rule, while Activity
#          condition of two rules are equal.

[subsumeContextCondition_14b1b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        equal(?actx ?acty)
                        ]

# 14.b.1.c one rule only has Location and Time Conditions, and another rule only has Time
#          condition with value subsuming the first rule, while Activity condition of two
#           rules are equal.

[subsumeContextCondition_14b1c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?actx ?acty)
                        ]


# 14.b.1.d one rule only has  Location and Time Conditions, and another rule don't have
#          Location and Time conditions, while Activity condition of two rules are equal.

[subsumeContextCondition_14b1d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?actx ?acty)
                        ]


# 14.b.2.a  Location and Activity conditions of two rules are with value subsumed, while
#           Time condition of two rules has equal value.
```

```
[subsumeContextCondition_14b2a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?timx ?timy)
                        ]

# 14.b.2.b one rule only has Location and Activity Conditions, and another rule only has
#          Location condition with value subsuming the first rule, while Time condition
#          of two rules are equal.

[subsumeContextCondition_14b2b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        equal(?timx ?timy)
                        ]


# 14.b.2.c one rule only has Location and Activity Conditions, and another rule only has
#          Activity condition with value subsuming the first rule, while Time condition
#          of two rules are equal.

[subsumeContextCondition_14b2c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?timx ?timy)
                        ]


# 14.b.2.d one rule only has Location and Activity Conditions, and another rule don't have
#        Location and Activity conditions, while Time condition of two rules are equal.

[subsumeContextCondition_14b2d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?timx ?timy)
                        ]


# 14.b.3.a  Time and Activity conditions of two rules are with value subsumed, while Locati
#           condition of two rules has equal value.
```

```
[subsumeContextCondition_14b3a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?lox ?loy)
                        ]

# 14.b.3.b one rule only has Time and Activity Conditions, and another rule only has Time
#          condition with value subsuming the first rule, while Location condition of two
#          rules are equal.

[subsumeContextCondition_14b3b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?lox ?loy)
                        ]

# 14.b.3.c one rule only has Time and Activity Conditions, and another rule only has
#          Activity condition with value subsuming the first rule, while Location
#          condition of two rules are equal.

[subsumeContextCondition_14b3c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?lox ?loy)
                        ]

# 14.b.3.d one rule only has Time and Activity Conditions, and another rule don't have Time
#           and Activity condition, while Location condition of two rules are equal.

[subsumeContextCondition_14b3d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?lox ?loy)
                        ]
```

```
# 14.c one rule only has Location, Time, and Activity Conditions, one of these conditions
#      is with value subsumed while other two conditions have equal value.


# 14.c.1  Location condition of two rules is with value subsumed, while Time and Activity
#         conditions of two rules have equal value.

[subsumeContextCondition_14c1: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?lox rdfs:subClassOf ?loy)
                          equal(?timx ?timy)
                          equal(?actx ?acty)
                          ]


# 14.c.2  Time condition of two rules is with value subsumed, while Location and Activity
#         conditions of two rules have equal value.

[subsumeContextCondition_14c2: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?timx rdfs:subClassOf ?timy)
                          equal(?lox ?loy)
                          equal(?actx ?acty)
                          ]


# 14.c.3  Activity condition of two rules is with value subsumed, while Location and Time
#         conditions of two rules have equal value.

[subsumeContextCondition_14c3: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          noValue(?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?actx rdfs:subClassOf ?acty)
                          equal(?lox ?loy)
                          equal(?timx ?timy)
                          ]


# 15.a both rules have all context conditions, and with value subsumed.
# this is not in micro version

# 15.b one rule have all four context condition, one of conditions has equal value, all oth
#      three subsumed (4)
# these four are not in micro version



# 15.c one rule have all four context condition, two of conditions have equal value, all ot
#      two subsumed (24)
```

```
# 15.c.1.a both rules have Time and Activity condition with equal value, and both rules hav
#          Target and Location Conditions with value subsumed.

[subsumeContextCondition_15c1a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          (?lox rdfs:subClassOf ?loy)
                          equal(?timx ?timy)
                          equal(?actx ?acty)
                          ]

# 15.c.1.b both rules have Time and Activity condition with equal value, one rule also has
#          Target and Location Conditions, and another rule only has Location condition
#          with value subsuming the first rule.

[subsumeContextCondition_15c1b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?lox rdfs:subClassOf ?loy)
                          equal(?timx ?timy)
                          equal(?actx ?acty)
                          ]


# 15.c.1.c  both rules have Time and Activity condition with equal value. one rule also
#           has Target and Location Conditions, and another rule only has Target condition
#           with value subsuming the first rule.

[subsumeContextCondition_15c1c: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          equal(?timx ?timy)
                          equal(?actx ?acty)
                          ]


# 15.c.1.d both rules have Time and Activity condition with equal value, one rule also has
#         Target and Location Conditions, and another rule don't have any context condition.

[subsumeContextCondition_15c1d: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
```

```
                        equal(?timx ?timy)
                        equal(?actx ?acty)
                        ]


# 15.c.2.a both rules have Location and Activity condition with equal value, and both rules
#          also have Target and Time Conditions with value subsumed.

[subsumeContextCondition_15c2a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?lox ?loy)
                        equal(?actx ?acty)
                        ]


#15.c.2.b both rules have Location and Activity condition with equal value, one rule alo ha
#         Target and Time Conditions, and another rule only has Time condition with value
#         subsuming the first rule.

[subsumeContextCondition_15c2b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?lox ?loy)
                        equal(?actx ?acty)
                        ]


#15.c.2.c both rules have Location and Activity condition with equal value, one rule alo
#         has Target and Time Conditions, one rule also has Target and Time Conditions,
#         and another rule only has Target condition with value subsuming the first rule.

[subsumeContextCondition_15c2c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        equal(?lox ?loy)
                        equal(?actx ?acty)
                        ]


#15.c.2.d both rules have Location and Activity condition with equal value, one rule also
#         has Target and Time Conditions, one rule also has Target and Time Conditions,
#         and another rule don't have any context condition.

[subsumeContextCondition_15c2d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
```

```
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?lox ?loy)
                        equal(?actx ?acty)
                        ]


# 15.c.3.a both rules have Location and Time condition with equal value, both rules also
#          have Target and Activity Conditions with value subsumed.

[subsumeContextCondition_15c3a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?lox ?loy)
                        equal(?timx ?timy)
                        ]


#15.c.3.b  both rules have Location and Time condition with equal value, one rule also has
#          Target and Activity Conditions, and another rule only has Activity condition
#          with value subsuming the first rule.

[subsumeContextCondition_15c3b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?lox ?loy)
                        equal(?timx ?timy)
                        ]


#15.c.3.c both rules have Location and Time condition with equal value, one rule also has
#         Target and Activity Conditions, and another rule only has Target condition with
#         value subsuming the first rule.

[subsumeContextCondition_15c3c: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        equal(?lox ?loy)
                        equal(?timx ?timy)
                        ]
```

```
#15.c.3.d both rules have Location and Time condition with equal value, one rule only has
#       Target and Activity Conditions, and another rule don't have any context condition.

[subsumeContextCondition_15c3d: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?lox ?loy)
                          equal(?timx ?timy)
                          ]


# 15.c.4.a both rules have Target and Activity condition with equal value, both rules also
#           have Location and Time Conditions with value subsumed.

[subsumeContextCondition_15c4a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?lox rdfs:subClassOf ?loy)
                          (?timx rdfs:subClassOf ?timy)
                          equal(?tax ?tay)
                          equal(?actx ?acty)
                          ]

# 15.c.4.b both rules have Target and Activity condition with equal value, one rule also
#          has Location and Time Conditions, and another rule only has Time condition
#          with value subsuming the first rule.

[subsumeContextCondition_15c4b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?timx rdfs:subClassOf ?timy)
                          equal(?tax ?tay)
                          equal(?actx ?acty)
                          ]


#15.c.4.c both rules have Target and Activity condition with equal value, one rule only
#         also Location and Time  Conditions, and another rule only has Location condition
#         with value subsuming the first rule.

[subsumeContextCondition_15c4c: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
```

```
                          (?lox rdfs:subClassOf ?loy)
                          equal(?tax ?tay)
                          equal(?actx ?acty)
                         ]


# 15.c.4.d both rules have Target and Activity condition with equal value, one rule also
#     has Location and Time Conditions, and another rule don't have any context condition.

[subsumeContextCondition_15c4d: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?actx ?acty)
                          ]


# 15.c.5.a both rules have Target and Time condition with equal value, both rules also have
#        Location and Activity Conditions with value subsumed.

[subsumeContextCondition_15c5a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?lox rdfs:subClassOf ?loy)
                          (?actx rdfs:subClassOf ?acty)
                          equal(?tax ?tay)
                          equal(?timx ?timy)
                          ]

# 15.c.5.b both rules have Target and Time condition with equal value, one rule also has
#          Location and Activity Conditions, and another rule only has Activity condition
#          with value subsuming the first rule.

[subsumeContextCondition_15c5b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?actx rdfs:subClassOf ?acty)
                          equal(?tax ?tay)
                          equal(?timx ?timy)
                          ]


# 15.c.5.c both rules have Target and Time condition with equal value, one rule also has
#          Location and Activity Conditions, and another rule only has Location condition
#          with value subsuming the first rule.

[subsumeContextCondition_15c5c: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
```

```
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        equal(?tax ?tay)
                        equal(?timx ?timy)
                        ]


# 15.c.5.d both rules have Target and Time condition with equal value, one rule also has
#       Location and Activity Conditions, and another rule don't have any context condition.

[subsumeContextCondition_15c5d: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        equal(?tax ?tay)
                        equal(?timx ?timy)
                        ]


#15.c.6.a both rules have Target and Location condition with equal value, both rules also h
#         Time and Activity Conditions with value subsumed.

[subsumeContextCondition_15c6a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?tax ?tay)
                        equal(?lox ?loy)
                        ]


# 15.c.6.b both rules have Target and Location condition with equal value, one rule also
#          has Time and Activity Conditions, and another rule only has Activity condition
#          with value subsuming the first rule.

[subsumeContextCondition_15c6b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?tax ?tay)
                        equal(?lox ?loy)
                        ]


#15.c.6.c both rules have Target and Location condition with equal value, one rule also has
#         Time and Activity Conditions, and another rule only has Time condition with value
```

```
#         subsuming the first rule.

[subsumeContextCondition_15c6c: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?timx rdfs:subClassOf ?timy)
                          equal(?tax ?tay)
                          equal(?lox ?loy)
                          ]


#15.c.6.d both rules have Target and Location condition with equal value, one rule also has
#         Time and Activity Conditions, and another rule don't have any context condition.

[subsumeContextCondition_15c6d: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?lox ?loy)
                          ]

# 15.d one rule have all four context condition, three of conditions have equal value,
#      another subsumed (8)

[subsumeContextCondition_15d1a: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          equal(?lox ?loy)
                          equal(?timx ?timy)
                          equal(?actx ?acty)
                          ]

[subsumeContextCondition_15d1b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          noValue(?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          equal(?lox ?loy)
                          equal(?timx ?timy)
                          equal(?actx ?acty)
                          ]
```

```
[subsumeContextCondition_15d2a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?tax ?tay)
                        (?lox rdfs:subClassOf ?loy)
                        equal(?timx ?timy)
                        equal(?actx ?acty)
                        ]

[subsumeContextCondition_15d2b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?tax ?tay)
                        equal(?timx ?timy)
                        equal(?actx ?acty)
                        ]


[subsumeContextCondition_15d3a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?tax ?tay)
                        equal(?lox ?loy)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?actx ?acty)
                        ]

[subsumeContextCondition_15d3b: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        equal(?tax ?tay)
                        equal(?lox ?loy)
                        equal(?actx ?acty)
                        ]


[subsumeContextCondition_15d4a: (?ccx pppl:subsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
```

```
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          (?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?lox ?loy)
                          equal(?timx ?timy)
                          (?actx rdfs:subClassOf ?acty)
                          ]

[subsumeContextCondition_15d4b: (?ccx pppl:subsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          (?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          equal(?tax ?tay)
                          equal(?lox ?loy)
                          equal(?timx ?timy)
                          ]


# ------------  The definition of mutuallySubsume_c & mutuallySubsumedBy_c (24)-----------

# 1. both rules only have Target and Location Conditions, and with value mutually subsumed.
[MS_ContextCondition1a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          (?loy rdfs:subClassOf ?lox)
                          ]

 [MS_ContextCondition1b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                          (?ccx pppl:hasLocation ?lox)
                          noValue(?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          (?ccy pppl:hasLocation ?loy)
                          noValue(?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?tay rdfs:subClassOf ?tax)
                          (?lox rdfs:subClassOf ?loy)
                          ]


# 2. both rules only have Target and Time Conditions, and with value mutually subsumed.

[MS_ContextCondition2a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                          (?ccx rdf:type pppl:ContextCondition)
                          (?ccy rdf:type pppl:ContextCondition)
                          (?ccx pppl:hasTarget ?tax)
                           noValue(?ccx pppl:hasLocation ?lox)
                          (?ccx pppl:hasTime ?timx)
                          noValue(?ccx pppl:hasActivity ?actx)
                          (?ccy pppl:hasTarget ?tay)
                          noValue(?ccy pppl:hasLocation ?loy)
                          (?ccy pppl:hasTime ?timy)
                          noValue(?ccy pppl:hasActivity ?acty)
                          (?tax rdfs:subClassOf ?tay)
                          (?timy rdfs:subClassOf ?timx)
                          ]
```

```
[MS_ContextCondition2b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                         noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        (?cy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?tay rdfs:subClassOf ?tax)
                        (?timx rdfs:subClassOf ?timy)
                        ]

# 3. both rules only have Target and Activity Conditions, and with value mutually subsumed.
[MS_ContextCondition3a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?acty rdfs:subClassOf ?actx)
                        ]

[MS_ContextCondition3b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tay rdfs:subClassOf ?tax)
                        (?actx rdfs:subClassOf ?acty)
                        ]

# 4. both rules only have Location and Time Conditions, and with value mutually subsumed.

[MS_ContextCondition4a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        (?timy rdfs:subClassOf ?timx)
                        ]

[MS_ContextCondition4b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        noValue(?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
```

```
                        noValue(?ccy pppl:hasActivity ?acty)
                        (?loy rdfs:subClassOf ?lox)
                        (?timx rdfs:subClassOf ?timy)
                        ]


# 5. both rules only have Location and Activity Conditions, and with value mutually subsume.

[MS_ContextCondition5a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        (?acty rdfs:subClassOf ?actx)
                        ]

[MS_ContextCondition5b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        noValue(?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        noValue(?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?loy rdfs:subClassOf ?lox)
                        (?actx rdfs:subClassOf ?acty)
                        ]

# 6. both rules only have Time and Activity Conditions, and with value mutually subsumed.

[MS_ContextCondition6a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?timx rdfs:subClassOf ?timy)
                        (?acty rdfs:subClassOf ?actx)
                        ]


[MS_ContextCondition6b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        noValue(?ccx pppl:hasTarget ?tax)
                        noValue(?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        noValue(?ccy pppl:hasTarget ?tay)
                        noValue(?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?timy rdfs:subClassOf ?timx)
                        (?actx rdfs:subClassOf ?acty)
                        ]


# 7. both rules have all four context conditions, Target and Location Conditions with value
#    mutually subsumed, and another two condition are equal.
```

```
[MS_ContextCondition7a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?loy rdfs:subClassOf ?lox)
                        equal(?timx ?timy)
                        equal(?actx ?acty)
                        ]

[MS_ContextCondition7b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tay rdfs:subClassOf ?tax)
                        (?lox rdfs:subClassOf ?loy)
                        equal(?timx ?timy)
                        equal(?actx ?acty)
                        ]


# 8. both rules have all four context conditions, Target and Time Conditions with value
#    mutually subsumed, and another two condition are equal.

[MS_ContextCondition8a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?timy rdfs:subClassOf ?timx)
                        equal(?lox ?loy)
                        equal(?actx ?acty)
                        ]

[MS_ContextCondition8b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?cy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tay rdfs:subClassOf ?tax)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?lox ?loy)
                        equal(?actx ?acty)
                        ]


# 9. both rules have all four context conditions, Target and Activity Conditions with value
#    mutually subsumed, and another two condition are equal.
```

```
[MS_ContextCondition9a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tax rdfs:subClassOf ?tay)
                        (?acty rdfs:subClassOf ?actx)
                        equal(?lox ?loy)
                        equal(?timx ?timy)
                        ]

[MS_ContextCondition9b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?tay rdfs:subClassOf ?tax)
                        (?actx rdfs:subClassOf ?acty)
                        equal(?lox ?loy)
                        equal(?timx ?timy)
                        ]


# 10. both rules have all four context conditions, Location and Time Conditions with value
#     mutually subsumed, and another two condition are equal.

[MS_ContextCondition10a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?lox rdfs:subClassOf ?loy)
                        (?timy rdfs:subClassOf ?timx)
                        equal(?actx ?acty)
                        equal(?tax ?tay)
                        ]

[MS_ContextCondition10b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                        (?ccx rdf:type pppl:ContextCondition)
                        (?ccy rdf:type pppl:ContextCondition)
                        (?ccx pppl:hasTarget ?tax)
                        (?ccx pppl:hasLocation ?lox)
                        (?ccx pppl:hasTime ?timx)
                        (?ccx pppl:hasActivity ?actx)
                        (?ccy pppl:hasTarget ?tay)
                        (?ccy pppl:hasLocation ?loy)
                        (?ccy pppl:hasTime ?timy)
                        (?ccy pppl:hasActivity ?acty)
                        (?loy rdfs:subClassOf ?lox)
                        (?timx rdfs:subClassOf ?timy)
                        equal(?actx ?acty)
                        equal(?tax ?tay)
                        ]


# 11. both rules have all four context conditions, Location and Activity Conditions with
#     value mutually subsumed, and another two condition are equal.
```

```
[MS_ContextCondition11a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         (?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         (?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         (?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         (?ccy pppl:hasActivity ?acty)
                         (?lox rdfs:subClassOf ?loy)
                         (?acty rdfs:subClassOf ?actx)
                         equal(?timx ?timy)
                         equal(?tax ?tay)
                         ]

[MS_ContextCondition11b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         (?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         (?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         (?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         (?ccy pppl:hasActivity ?acty)
                         (?loy rdfs:subClassOf ?lox)
                         (?actx rdfs:subClassOf ?acty)
                         equal(?timx ?timy)
                         equal(?tax ?tay)
                         ]


# 12. both rules have all four context conditions, Time and Activity Conditions with value
#     mutually subsumed, and another two condition are equal.

[MS_ContextCondition12a: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         (?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         (?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         (?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         (?ccy pppl:hasActivity ?acty)
                         (?timx rdfs:subClassOf ?timy)
                         (?acty rdfs:subClassOf ?actx)
                         equal(?lox ?loy)
                         equal(?tax ?tay)
                         ]


[MS_ContextCondition12b: (?ccx pppl:mutuallySubsumedBy_c ?ccy) <-
                         (?ccx rdf:type pppl:ContextCondition)
                         (?ccy rdf:type pppl:ContextCondition)
                         (?ccx pppl:hasTarget ?tax)
                         (?ccx pppl:hasLocation ?lox)
                         (?ccx pppl:hasTime ?timx)
                         (?ccx pppl:hasActivity ?actx)
                         (?ccy pppl:hasTarget ?tay)
                         (?ccy pppl:hasLocation ?loy)
                         (?ccy pppl:hasTime ?timy)
                         (?ccy pppl:hasActivity ?acty)
                         (?timy rdfs:subClassOf ?timx)
                         (?actx rdfs:subClassOf ?acty)
                         equal(?lox ?loy)
                         equal(?tax ?tay)
                         ]
```

Appendix F: A full list of Jena inference rules used to conduct policy evaluation[34]

```
# Policy Evaluation Inference Rules
@prefix pppl: <http://www.ee.ucl.ac.uk/~jezhang/privacypreferenceruleontology#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.

# @include <RDFS>.
# @include<OWL>.

# ----------  Rules to find fully matched preference rules that are applicable to evaluate
# Data Collecting Policy (8)-------------------

[fullyMatchRule1: (?rulex pppl:fullyMatch ?ruley) <-
                  (?rulex rdf:type pppl:Rule)
                  (?ruley rdf:type pppl:Rule)
                  (?rulex pppl:hasData ?dx)
                  (?ruley pppl:hasData ?dy)
                  equal(?dx ?dy)
                  (?rulex pppl:hasPolicyCon ?pcx)
                  (?ruley pppl:hasPolicyCon ?pcy)
                  (?pcx pppl:subsumedBy_p ?pcy)
                  (?rulex pppl:hasContextCon ?ccx)
                  (?ccx pppl:hasTarget ?tax)
                  (?ruley pppl:hasContextCon ?ccy)
                  (?ccy pppl:hasTarget ?tay)
                  equal(?tax ?tay)
                  ]

[fullyMatchRule1a: (?rulex pppl:fullyMatch ?ruley) <-
                  (?rulex rdf:type pppl:Rule)
                  (?ruley rdf:type pppl:Rule)
                  (?rulex pppl:hasData ?dx)
                  (?ruley pppl:hasData ?dy)
                  equal(?dx ?dy)
                  (?rulex pppl:hasPolicyCon ?pcx)
                  (?ruley pppl:hasPolicyCon ?pcy)
                  (?pcx pppl:subsumedBy_p ?pcy)
                  (?rulex pppl:hasContextCon ?ccx)
                  noValue(?ccx pppl:hasTarget ?tax)
                  (?ruley pppl:hasContextCon ?ccy)
                  noValue(?ccy pppl:hasTarget ?tay)
                  ]

[fullyMatchRule2: (?rulex pppl:fullyMatch ?ruley) <-
                  (?rulex rdf:type pppl:Rule)
                  (?ruley rdf:type pppl:Rule)
                  (?rulex pppl:hasData ?dx)
                  (?ruley pppl:hasData ?dy)
                  (?dx rdfs:subClassOf ?dy)
                  (?rulex pppl:hasPolicyCon ?pcx)
                  (?ruley pppl:hasPolicyCon ?pcy)
                  (?pcx pppl:subsumedBy_p ?pcy)
                  (?rulex pppl:hasContextCon ?ccx)
                  (?ccx pppl:hasTarget ?tax)
                  (?ruley pppl:hasContextCon ?ccy)
                  (?ccy pppl:hasTarget ?tay)
                  equal(?tax ?tay)
                  ]

[fullyMatchRule2a: (?rulex pppl:fullyMatch ?ruley) <-
                  (?rulex rdf:type pppl:Rule)
                  (?ruley rdf:type pppl:Rule)
                  (?rulex pppl:hasData ?dx)
                  (?ruley pppl:hasData ?dy)
                  (?dx rdfs:subClassOf ?dy)
                  (?rulex pppl:hasPolicyCon ?pcx)
                  (?ruley pppl:hasPolicyCon ?pcy)
                  (?pcx pppl:subsumedBy_p ?pcy)
                  (?rulex pppl:hasContextCon ?ccx)
                  noValue(?ccx pppl:hasTarget ?tax)
                  (?ruley pppl:hasContextCon ?ccy)
                  noValue(?ccy pppl:hasTarget ?tay)
                  ]
```

[34] Section 6.5.1 describes briefly the development of these inference rules. The list of the inference rules is also available at http://ee.ucl.ac.uk/~jezhang/policyevaluationrules.rules

```
[fullyMatchRule3: (?rulex pppl:fullyMatch ?ruley) <-
                  (?rulex rdf:type pppl:Rule)
                  (?ruley rdf:type pppl:Rule)
                  (?rulex pppl:hasData ?dx)
                  (?ruley pppl:hasData ?dy)
                   equal(?dx ?dy)
                  (?rulex pppl:hasPolicyCon ?pcx)
                  (?ruley pppl:hasPolicyCon ?pcy)
                  (?pcx pppl:subsumedBy_p ?pcy)
                  (?rulex pppl:hasContextCon ?ccx)
                  (?ccx pppl:hasTarget ?tax)
                  (?ruley pppl:hasContextCon ?ccy)
                  (?ccy pppl:hasTarget ?tay)
                  (?tax rdfs:subClassOf ?tay)
                  ]

[fullyMatchRule3a: (?rulex pppl:fullyMatch ?ruley) <-
                  (?rulex rdf:type pppl:Rule)
                  (?ruley rdf:type pppl:Rule)
                  (?rulex pppl:hasData ?dx)
                  (?ruley pppl:hasData ?dy)
                   equal(?dx ?dy)
                  (?rulex pppl:hasPolicyCon ?pcx)
                  (?ruley pppl:hasPolicyCon ?pcy)
                  (?pcx pppl:subsumedBy_p ?pcy)
                  (?rulex pppl:hasContextCon ?ccx)
                  (?ccx pppl:hasTarget ?tax)
                  (?ruley pppl:hasContextCon ?ccy)
                  noValue(?ccy pppl:hasTarget ?tay)
                  ]

[fullyMatchRule4: (?rulex pppl:fullyMatch ?ruley) <-
                  (?rulex rdf:type pppl:Rule)
                  (?ruley rdf:type pppl:Rule)
                  (?rulex pppl:hasData ?dx)
                  (?ruley pppl:hasData ?dy)
                  (?dx rdfs:subClassOf ?dy)
                  (?rulex pppl:hasPolicyCon ?pcx)
                  (?ruley pppl:hasPolicyCon ?pcy)
                  (?pcx pppl:subsumedBy_p ?pcy)
                  (?rulex pppl:hasContextCon ?ccx)
                  (?ccx pppl:hasTarget ?tax)
                  (?ruley pppl:hasContextCon ?ccy)
                  (?ccy pppl:hasTarget ?tay)
                  (?tax rdfs:subClassOf ?tay)
                  ]

[fullyMatchRule4a: (?rulex pppl:fullyMatch ?ruley) <-
                  (?rulex rdf:type pppl:Rule)
                  (?ruley rdf:type pppl:Rule)
                  (?rulex pppl:hasData ?dx)
                  (?ruley pppl:hasData ?dy)
                  (?dx rdfs:subClassOf ?dy)
                  (?rulex pppl:hasPolicyCon ?pcx)
                  (?ruley pppl:hasPolicyCon ?pcy)
                  (?pcx pppl:subsumedBy_p ?pcy)
                  (?rulex pppl:hasContextCon ?ccx)
                  (?ccx pppl:hasTarget ?tax)
                  (?ruley pppl:hasContextCon ?ccy)
                  noValue(?ccy pppl:hasTarget ?tay)
                  ]



# ----------  Rules to find partially matched rules that are applicable to evaluate Data
# Collecting Policy (20)------------------

[partiallyMatchRule1a:(?rulex pppl:partiallyMatch ?ruley) <-
                      (?preference rdf:type pppl:Preferences)
                      (?preference pppl:hasMetaPolicy ?metapolicy)
                      equal(?metapolicy pppl:Optimistic)
                      (?rulex rdf:type pppl:Rule)
                      (?ruley rdf:type pppl:Rule)
                      (?ruley pppl:hasBehavior ?by)
                      equal(?by pppl:Permit)
                      (?rulex pppl:hasData ?dx)
```

```
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        equal(?tax ?tay)
                        ]


[partiallyMatchRule1b: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        equal(?tax ?tay)
                        ]


 [partiallyMatchRule1c: (?rulex pppl:partiallyMatch ?ruley) <-
                        (?preference rdf:type pppl:Preferences)
                        (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Optimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Permit)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        noValue(?ccy pppl:hasTarget ?tay)
                        ]


 [partiallyMatchRule1d: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        noValue(?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        noValue(?ccy pppl:hasTarget ?tay)
                        ]
```

```
[partiallyMatchRule2a: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Optimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                       equal(?by pppl:Permit)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                       equal(?dx ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        (?tay rdfs:subClassOf ?tax)
                        ]


[partiallyMatchRule2b: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                       equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                       equal(?dx ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        (?tay rdfs:subClassOf ?tax)
                        ]


[partiallyMatchRule2c: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Optimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                       equal(?by pppl:Permit)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                       equal(?dx ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                       noValue(?ccy pppl:hasTarget ?tay)
                       ]


[partiallyMatchRule2d: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                       equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
```

```
                        equal(?dx ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        noValue(?ccy pppl:hasTarget ?tay)
                        ]


[partiallyMatchRule3a: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Optimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Permit)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        (?tay rdfs:subClassOf ?tax)
                        ]


[partiallyMatchRule3b: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        (?tay rdfs:subClassOf ?tax)
                        ]


[partiallyMatchRule3c: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Optimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Permit)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        noValue(?ccy pppl:hasTarget ?tay)
                        ]
```

```
[partiallyMatchRule3d: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                       equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                       noValue(?ccy pppl:hasTarget ?tay)
                        ]


[partiallyMatchRule4a: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Optimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                       equal(?by pppl:Permit)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        (?tax rdfs:subClassOf ?tay)
                        ]


[partiallyMatchRule4b: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                       equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        (?tax rdfs:subClassOf ?tay)
                        ]


[partiallyMatchRule4c: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Optimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                       equal(?by pppl:Permit)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
```

```
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        noValue(?ccy pppl:hasTarget ?tay)
                        ]


[partiallyMatchRule4d: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dy rdfs:subClassOf ?dx)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        noValue(?ccy pppl:hasTarget ?tay)
                        ]


[partiallyMatchRule5a: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Optimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Permit)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dx rdfs:subClassOf ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        (?tay rdfs:subClassOf ?tax)
                        ]


[partiallyMatchRule5b: (?rulex pppl:partiallyMatch ?ruley) <-
                       (?preference rdf:type pppl:Preferences)
                       (?preference pppl:hasMetaPolicy ?metapolicy)
                        equal(?metapolicy pppl:Pessimistic)
                        (?rulex rdf:type pppl:Rule)
                        (?ruley rdf:type pppl:Rule)
                        (?ruley pppl:hasBehavior ?by)
                        equal(?by pppl:Forbid)
                        (?rulex pppl:hasData ?dx)
                        (?ruley pppl:hasData ?dy)
                        (?dx rdfs:subClassOf ?dy)
                        (?rulex pppl:hasPolicyCon ?pcx)
                        (?ruley pppl:hasPolicyCon ?pcy)
                        (?pcx pppl:subsumedBy_p ?pcy)
                        (?rulex pppl:hasContextCon ?ccx)
                        (?ccx pppl:hasTarget ?tax)
                        (?ruley pppl:hasContextCon ?ccy)
                        (?ccy pppl:hasTarget ?tay)
                        (?tay rdfs:subClassOf ?tax)
                        ]


[partiallyMatchRule5c: (?rulex pppl:partiallyMatch ?ruley) <-
```

```
                    (?preference rdf:type pppl:Preferences)
                    (?preference pppl:hasMetaPolicy ?metapolicy)
                     equal(?metapolicy pppl:Optimistic)
                     (?rulex rdf:type pppl:Rule)
                     (?ruley rdf:type pppl:Rule)
                     (?ruley pppl:hasBehavior ?by)
                     equal(?by pppl:Permit)
                     (?rulex pppl:hasData ?dx)
                     (?ruley pppl:hasData ?dy)
                     (?dx rdfs:subClassOf ?dy)
                     (?rulex pppl:hasPolicyCon ?pcx)
                     (?ruley pppl:hasPolicyCon ?pcy)
                     (?pcx pppl:subsumedBy_p ?pcy)
                     (?rulex pppl:hasContextCon ?ccx)
                     (?ccx pppl:hasTarget ?tax)
                     (?ruley pppl:hasContextCon ?ccy)
                     noValue(?ccy pppl:hasTarget ?tay)
                     ]

[partiallyMatchRule5d: (?rulex pppl:partiallyMatch ?ruley) <-
                    (?preference rdf:type pppl:Preferences)
                    (?preference pppl:hasMetaPolicy ?metapolicy)
                     equal(?metapolicy pppl:Pessimistic)
                     (?rulex rdf:type pppl:Rule)
                     (?ruley rdf:type pppl:Rule)
                     (?ruley pppl:hasBehavior ?by)
                     equal(?by pppl:Forbid)
                     (?rulex pppl:hasData ?dx)
                     (?ruley pppl:hasData ?dy)
                     (?dx rdfs:subClassOf ?dy)
                     (?rulex pppl:hasPolicyCon ?pcx)
                     (?ruley pppl:hasPolicyCon ?pcy)
                     (?pcx pppl:subsumedBy_p ?pcy)
                     (?rulex pppl:hasContextCon ?ccx)
                     (?ccx pppl:hasTarget ?tax)
                     (?ruley pppl:hasContextCon ?ccy)
                     noValue(?ccy pppl:hasTarget ?tay)
                     ]


# ------------  The definition of subsumedBy_p (27 situations, complete) ----------------

#1. both rules do not have any policy conditions

[subsumedPolicyCondition1:  (?pcx pppl:subsumedBy_p ?pcy)  <-
                              (?pcx rdf:type pppl:PolicyCondition)
                              (?pcy rdf:type pppl:PolicyCondition)
                               noValue(?pcx pppl:hasPurpose ?purx)
                               noValue(?pcx pppl:hasRecipient ?recx)
                               noValue(?pcx pppl:hasRetention ?retx)
                               noValue(?pcy pppl:hasPurpose ?pury)
                               noValue(?pcy pppl:hasRecipient ?recy)
                               noValue(?pcy pppl:hasRetention ?rety)
                               ]


# 2. both rules only have Purpose Condition, and with value equal or subsumedby.

[subsumedPolicyCondition2a: (?pcx pppl:subsumedBy_p ?pcy) <-
                             (?pcx rdf:type pppl:PolicyCondition)
                             (?pcy rdf:type pppl:PolicyCondition)
                             (?pcx pppl:hasPurpose ?purx)
                              noValue(?pcx pppl:hasRecipient ?recx)
                              noValue(?pcx pppl:hasRetention ?retx)
                             (?pcy pppl:hasPurpose ?pury)
                              noValue(?pcy pppl:hasRecipient ?recy)
                              noValue(?pcy pppl:hasRetention ?rety)
                              equal(?purx ?pury)
                             ]

[subsumePolicyCondition2b: (?pcx pppl:subsumedBy_p ?pcy) <-
                           (?pcx rdf:type pppl:PolicyCondition)
                           (?pcy rdf:type pppl:PolicyCondition)
```

```
                          (?pcx pppl:hasPurpose ?purx)
                          noValue(?pcx pppl:hasRecipient ?recx)
                          noValue(?pcx pppl:hasRetention ?retx)
                          noValue(?pcy pppl:hasPurpose ?pury)
                          noValue(?pcy pppl:hasRecipient ?recy)
                          noValue(?pcy pppl:hasRetention ?rety)
                          ]


# 3. both rules only have Recipient Condition, and with value equal or subsumedby.

[subsumedPolicyCondition3a: (?pcx pppl:subsumedBy_p ?pcy) <-
                            (?pcx rdf:type pppl:PolicyCondition)
                            (?pcy rdf:type pppl:PolicyCondition)
                             noValue(?pcx pppl:hasPurpose ?purx)
                             (?pcx pppl:hasRecipient ?recx)
                             noValue(?pcx pppl:hasRetention ?retx)
                             noValue(?pcy pppl:hasPurpose ?pury)
                             (?pcy pppl:hasRecipient ?recy)
                             noValue(?pcy pppl:hasRetention ?rety)
                             equal(?recx ?recy)
                            ]

[subsumedPolicyCondition3b:  (?pcx pppl:subsumedBy_p ?pcy) <-
                            (?pcx rdf:type pppl:PolicyCondition)
                            (?pcy rdf:type pppl:PolicyCondition)
                             noValue(?pcx pppl:hasPurpose ?purx)
                             (?pcx pppl:hasRecipient ?recx)
                             noValue(?pcx pppl:hasRetention ?retx)
                             noValue(?pcy pppl:hasPurpose ?pury)
                             noValue(?pcy pppl:hasRecipient ?recy)
                             noValue(?pcy pppl:hasRetention ?rety)
                            ]


# 4. both rules only have Retention Condition, and with value equal or subsumedby.

[subsumedPolicyCondition4a: (?pcx pppl:subsumedBy_p ?pcy) <-
                            (?pcx rdf:type pppl:PolicyCondition)
                            (?pcy rdf:type pppl:PolicyCondition)
                             noValue(?pcx pppl:hasPurpose ?purx)
                             noValue(?pcx pppl:hasRecipient ?recx)
                             (?pcx pppl:hasRetention ?retx)
                             noValue(?pcy pppl:hasPurpose ?pury)
                             noValue(?pcy pppl:hasRecipient ?recy)
                             (?pcy pppl:hasRetention ?rety)
                             equal(?retx ?rety)
                            ]

[subsumedPolicyCondition4b: (?pcx pppl:subsumedBy_p ?pcy) <-
                            (?pcx rdf:type pppl:PolicyCondition)
                            (?pcy rdf:type pppl:PolicyCondition)
                             noValue(?pcx pppl:hasPurpose ?purx)
                             noValue(?pcx pppl:hasRecipient ?recx)
                             (?pcx pppl:hasRetention ?retx)
                             noValue(?pcy pppl:hasPurpose ?pury)
                             noValue(?pcy pppl:hasRecipient ?recy)
                             noValue(?pcy pppl:hasRetention ?rety)
                            ]


# 5. both rules only have Purpose and Recipient Condition, and with value equal or
#    subsumedby.

[subsumedPolicyCondition5a: (?pcx pppl:subsumedBy_p ?pcy) <-
                            (?pcx rdf:type pppl:PolicyCondition)
                            (?pcy rdf:type pppl:PolicyCondition)
                             (?pcx pppl:hasPurpose ?purx)
                             (?pcx pppl:hasRecipient ?recx)
                             noValue(?pcx pppl:hasRetention ?retx)
                             (?pcy pppl:hasPurpose ?pury)
                             (?pcy pppl:hasRecipient ?recy)
                             noValue(?pcy pppl:hasRetention ?rety)
```

```
                                equal(?purx ?pury)
                                equal(?recx ?recy)
                                ]

[subsumedPolicyCondition5b: (?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                noValue(?pcx pppl:hasRetention ?retx)
                                noValue(?pcy pppl:hasPurpose ?pury)
                                (?pcy pppl:hasRecipient ?recy)
                                noValue(?pcy pppl:hasRetention ?rety)
                                equal(?recx ?recy)
                                ]

[subsumedPolicyCondition5c:   (?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                noValue(?pcx pppl:hasRetention ?retx)
                                (?pcy pppl:hasPurpose ?pury)
                                noValue(?pcy pppl:hasRecipient ?recy)
                                noValue(?pcy pppl:hasRetention ?rety)
                                equal(?purxx ?pury)
                                ]

[subsumedPolicyCondition5d: (?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                noValue(?pcx pppl:hasRetention ?retx)
                                noValue(?pcy pppl:hasPurpose ?pury)
                                noValue(?pcy pppl:hasRecipient ?recy)
                                noValue(?pcy pppl:hasRetention ?rety)
                                ]


# 6. both rules only have Purpose and Retention Condition, and with value equal or
#    subsumedby.

[subsumedPolicyCondition6a: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                noValue(?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                (?pcy pppl:hasPurpose ?pury)
                                noValue(?pcy pppl:hasRecipient ?recy)
                                (?pcy pppl:hasRetention ?rety)
                                equal(?purx ?pury)
                                equal(?retx ?rety)
                                ]

[subsumedPolicyCondition6b: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                noValue(?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                noValue(?pcy pppl:hasPurpose ?pury)
                                noValue(?pcy pppl:hasRecipient ?recy)
                                (?pcy pppl:hasRetention ?rety)
                                equal(?retx ?rety)
                                ]

[subsumedPolicyCondition6c: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                noValue(?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                (?pcy pppl:hasPurpose ?pury)
```

```
                                noValue(?pcy pppl:hasRecipient ?recy)
                                noValue(?pcy pppl:hasRetention ?rety)
                                equal(?purx ?pury)
                                ]

[subsumedPolicyCondition6d: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                noValue(?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                noValue(?pcy pppl:hasPurpose ?pury)
                                noValue(?pcy pppl:hasRecipient ?recy)
                                noValue(?pcy pppl:hasRetention ?rety)
                                ]


# 7. both rules only have Recipient and Retention Condition, and with value equal or
#    subsumedby.

[subsumedPolicyCondition7a: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                              (?pcx rdf:type pppl:PolicyCondition)
                              (?pcy rdf:type pppl:PolicyCondition)
                               noValue(?pcx pppl:hasPurpose ?purx)
                               (?pcx pppl:hasRecipient ?recx)
                               (?pcx pppl:hasRetention ?retx)
                               noValue(?pcy pppl:hasPurpose ?pury)
                               (?pcy pppl:hasRecipient ?recy)
                               (?pcy pppl:hasRetention ?rety)
                               equal(?recx ?recy)
                               equal(?retx ?rety)
                               ]

[subsumedPolicyCondition7b: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                              (?pcx rdf:type pppl:PolicyCondition)
                              (?pcy rdf:type pppl:PolicyCondition)
                               noValue(?pcx pppl:hasPurpose ?purx)
                               (?pcx pppl:hasRecipient ?recx)
                               (?pcx pppl:hasRetention ?retx)
                               noValue(?pcy pppl:hasPurpose ?pury)
                               noValue(?pcy pppl:hasRecipient ?recy)
                               (?pcy pppl:hasRetention ?rety)
                               equal(?retx ?rety)
                               ]

[subsumedPolicyCondition7c: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                              (?pcx rdf:type pppl:PolicyCondition)
                              (?pcy rdf:type pppl:PolicyCondition)
                               noValue(?pcx pppl:hasPurpose ?purx)
                               (?pcx pppl:hasRecipient ?recx)
                               (?pcx pppl:hasRetention ?retx)
                               noValue(?pcy pppl:hasPurpose ?pury)
                               (?pcy pppl:hasRecipient ?recy)
                               noValue(?pcy pppl:hasRetention ?rety)
                               equal(?recx ?recy)
                               ]

[subsumedPolicyCondition7d: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                              (?pcx rdf:type pppl:PolicyCondition)
                              (?pcy rdf:type pppl:PolicyCondition)
                               noValue(?pcx pppl:hasPurpose ?purx)
                               (?pcx pppl:hasRecipient ?recx)
                               (?pcx pppl:hasRetention ?retx)
                               noValue(?pcy pppl:hasPurpose ?pury)
                               noValue(?pcy pppl:hasRecipient ?recy)
                               noValue(?pcy pppl:hasRetention ?rety)
                               ]


# 8. both rules have all policy, and with value equal or subsumedby.

[subsumedPolicyCondition8a: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                              (?pcx rdf:type pppl:PolicyCondition)
                              (?pcy rdf:type pppl:PolicyCondition)
```

```
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                (?pcy pppl:hasPurpose ?pury)
                                (?pcy pppl:hasRecipient ?recy)
                                (?pcy pppl:hasRetention ?rety)
                                equal(?purx ?pury)
                                equal(?recx ?recy)
                                equal(?retx ?rety)
                                ]

[subsumedPolicyCondition8b: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                noValue(?pcy pppl:hasPurpose ?pury)
                                (?pcy pppl:hasRecipient ?recy)
                                (?pcy pppl:hasRetention ?rety)
                                equal(?recx ?recy)
                                equal(?retx ?rety)
                                ]

[subsumedPolicyCondition8c: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                (?pcy pppl:hasPurpose ?pury)
                                noValue(?pcy pppl:hasRecipient ?recy)
                                (?pcy pppl:hasRetention ?rety)
                                equal(?purx ?pury)
                                equal(?retx ?rety)
                                ]

[subsumedPolicyCondition8d: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                (?pcy pppl:hasPurpose ?pury)
                                (?pcy pppl:hasRecipient ?recy)
                                noValue(?pcy pppl:hasRetention ?rety)
                                equal(?purx ?pury)
                                equal(?recx ?recy)
                                ]

[subsumedPolicyCondition8e: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                noValue(?pcy pppl:hasPurpose ?pury)
                                noValue(?pcy pppl:hasRecipient ?recy)
                                (?pcy pppl:hasRetention ?rety)
                                equal(?retx ?rety)
                                ]

[subsumedPolicyCondition8f: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
                               (?pcy rdf:type pppl:PolicyCondition)
                                (?pcx pppl:hasPurpose ?purx)
                                (?pcx pppl:hasRecipient ?recx)
                                (?pcx pppl:hasRetention ?retx)
                                noValue(?pcy pppl:hasPurpose ?pury)
                                (?pcy pppl:hasRecipient ?recy)
                                noValue(?pcy pppl:hasRetention ?rety)
                                equal(?recx ?recy)
                                ]

[subsumedPolicyCondition8g: ( ?pcx pppl:subsumedBy_p ?pcy) <-
                               (?pcx rdf:type pppl:PolicyCondition)
```

```
                                (?pcy rdf:type pppl:PolicyCondition)
                                 (?pcx pppl:hasPurpose ?purx)
                                 (?pcx pppl:hasRecipient ?recx)
                                 (?pcx pppl:hasRetention ?retx)
                                 (?pcy pppl:hasPurpose ?pury)
                                 noValue(?pcy pppl:hasRecipient ?recy)
                                 noValue(?pcy pppl:hasRetention ?rety)
                                 equal(?purx ?pury)
                                 ]

[subsumedPolicyCondition8h:  ( ?pcx pppl:subsumedBy_p ?pcy) <-
                                (?pcx rdf:type pppl:PolicyCondition)
                                (?pcy rdf:type pppl:PolicyCondition)
                                 (?pcx pppl:hasPurpose ?purx)
                                 (?pcx pppl:hasRecipient ?recx)
                                 (?pcx pppl:hasRetention ?retx)
                                 noValue(?pcy pppl:hasPurpose ?pury)
                                 noValue(?pcy pppl:hasRecipient ?recy)
                                 noValue(?pcy pppl:hasRetention ?rety)
                                 ]
```

Appendix G: A full list of Jena inference rules used to conduct context reasoning

```
# Context Reasoning Inference Rules File
@prefix per: <http://www.ee.ucl.ac.uk/~jezhang/PersonalInformationOntology#>.
@prefix loc: <http://www.ee.ucl.ac.uk/~jezhang/LocationOntology#>.
@prefix tme: <http://www.ee.ucl.ac.uk/~jezhang/TimeOntology#>.
@prefix act: <http://www.ee.ucl.ac.uk/~jezhang/ActivityOntology#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.

# @include <RDFS>.
# @include<OWL>.

# -----------  Inference rules to define spatial reasoning (14) ------------------

[SpatialReasoning1:(?x loc:locatedIn ?y) <-
                   (?x rdf:type loc:ContextualLocation)
                   (?y rdf:type loc:PremiseLocation)
                   (?x loc:hasGeographicAddress ?x1)
                   (?x1 rdf:type loc:GeographicAddress)
                   (?y loc:hasCoordinate ?y1)
                   (?y1 rdf:type loc:Coordinate)
                   (?x1 loc:relatedTo ?y1)
                   ]

[SpatialReasoning2:(?x loc:locatedIn ?y) <-
                   (?x rdf:type loc:ContextualLocation)
                   (?y rdf:type loc:PremiseLocation)
                   (?x loc:hasCoordinate ?x1)
                   (?x1 rdf:type loc:Coordinate)
                   (?y loc:hasGeographicAddress ?y1)
                   (?y1 rdf:type loc:GeographicAddress)
                   (?x1 loc:relatedTo ?y1)
                   ]

[SpatialReasoning3:(?x loc:locatedIn ?y) <-
                   (?x rdf:type loc:ContextualLocation)
                   (?y rdf:type loc:PremiseLocation)
                   (?x loc:hasCoordinate ?x1)
                   (?x1 rdf:type loc:Coordinate)
                   (?y loc:hasNetworkAddress ?y1)
                   (?y1 rdf:type loc:NetworkAddress)
                   (?x1 loc:relatedTo ?y1)
                   ]

[SpatialReasoning4:(?x loc:locatedIn ?y) <-
                   (?x rdf:type loc:ContextualLocation)
                   (?y rdf:type loc:PremiseLocation)
                   (?x loc:hasNetworkAddress ?x1)
                   (?x1 rdf:type loc:NetworkAddress)
                   (?y loc:hasCoordinate ?y1)
                   (?y1 rdf:type loc:Coordinate)
                   (?x1 loc:relatedTo ?y1)
                   ]

[SpatialReasoning5:(?x loc:locatedIn ?y) <-
                   (?x rdf:type loc:ContextualLocation)
                   (?y rdf:type loc:PremiseLocation)
                   (?x loc:hasGeographicAddress ?x1)
                   (?x1 rdf:type loc:GeographicAddress)
                   (?y loc:hasNetworkAddress ?y1)
                   (?y1 rdf:type loc:NetworkAddress)
                   (?x1 loc:relatedTo ?y1)
                   ]
```

```
[SpatialReasoning6: (?x loc:locatedIn ?y) <-
                    (?x rdf:type loc:ContextualLocation)
                    (?y rdf:type loc:PremiseLocation)
                    (?x loc:hasNetworkAddress ?x1)
                    (?x1 rdf:type loc:NetworkAddress)
                    (?y loc:hasGeographicAddress ?y1)
                    (?y1 rdf:type loc:GeographicAddress)
                    (?x1 loc:relatedTo ?y1)
                    ]


[SpatialReasoning7:  (?x loc:disconnectedWith ?y)<-
                     (?x rdf:type loc:ContextualLocation)
                     (?y rdf:type loc:PremiseLocation)
                     (?x loc:hasGeographicAddress ?x1)
                     (?x1 rdf:type loc:GeographicAddress)
                     (?y loc:hasCoordinate ?y1)
                     (?y1 rdf:type loc:Coordinate)
                     noValue(?x1 loc:relatedTo ?y1)
                     ]

[SpatialReasoning8: (?x loc:disconnectedWith ?y) <-
                    (?x rdf:type loc:ContextualLocation)
                    (?y rdf:type loc:PremiseLocation)
                    (?x loc:hasCoordinate ?x1)
                    (?x1 rdf:type loc:Coordinate)
                    (?y loc:hasGeographicAddress ?y1)
                    (?y1 rdf:type loc:GeographicAddress)
                     noValue(?x1 loc:relatedTo ?y1)
                    ]

[SpatialReasoning9: (?x loc:disconnectedWith ?y) <-
                    (?x rdf:type loc:ContextualLocation)
                    (?y rdf:type loc:PremiseLocation)
                    (?x loc:hasCoordinate ?x1)
                    (?x1 rdf:type loc:Coordinate)
                    (?y loc:hasNetworkAddress ?y1)
                    (?y1 rdf:type loc:NetworkAddress)
                     noValue(?x1 loc:relatedTo ?y1)
                    ]

[SpatialReasoning10: (?x loc:disconnectedWith ?y) <-
                    (?x rdf:type loc:ContextualLocation)
                    (?y rdf:type loc:PremiseLocation)
                    (?x loc:hasNetworkAddress ?x1)
                    (?x1 rdf:type loc:NetworkAddress)
                    (?y loc:hasCoordinate ?y1)
                    (?y1 rdf:type loc:Coordinate)
                     noValue(?x1 loc:relatedTo ?y1)
                    ]

[SpatialReasoning11: (?x loc:disconnectedWith ?y) <-
                    (?x rdf:type loc:ContextualLocation)
                    (?y rdf:type loc:PremiseLocation)
                    (?x loc:hasGeographicAddress ?x1)
                    (?x1 rdf:type loc:GeographicAddress)
                    (?y loc:hasNetworkAddress ?y1)
                    (?y1 rdf:type loc:NetworkAddress)
                     noValue(?x1 loc:relatedTo ?y1)
                    ]


[SpatialReasoning12: (?x loc:disconnectedWith ?y) <-
                    (?x rdf:type loc:ContextualLocation)
                    (?y rdf:type loc:PremiseLocation)
                    (?x loc:hasNetworkAddress ?x1)
                    (?x1 rdf:type loc:NetworkAddress)
                    (?y loc:hasGeographicAddress ?y1)
                    (?y1 rdf:type loc:GeographicAddress)
                     noValue(?x1 loc:relatedTo ?y1)
                    ]


[SpatialReasoning13: (?x loc:spatiallySubsume ?y) <-
                    (?x rdf:type loc:GeographicAddress)
                    (?x rdf:type loc:GeographicAddress)
```

```
                    (?x loc:relatedTo ?y)
                    (?x loc:hasDisclosureLevel ?xd)
                    (?y loc:hasDisclosureLevel ?yd)
                    greaterThan(?xd ?yd)
                    ]

[SpatialReasoning14:(?x loc:spatiallySubsume ?y) <-
                    (?x rdf:type loc:SymbolicAddressInFixedStructure)
                    (?x rdf:type loc:SymbolicAddressInFixedStructure)
                    (?x loc:relatedTo ?y)
                    (?x loc:hasDisclosureLevel ?xd)
                    (?y loc:hasDisclosureLevel ?yd)
                    greaterThan(?xd ?yd)
                    ]


# ----------- Inference rules to define temporal reasoning (12) ------------------

[TemporalReasoning1:(?x tme:insEqual ?y) <-
                     (?x rdf:type tme:InstantEntity)
                     (?y rdf:type tme:InstantEntity)
                     (?x tme:is ?isX)
                     (?isX rdf:type tme:XSDDateTime)
                     (?y tme:is ?isY)
                     (?isY rdf:type tme:XSDDateTime)
                     equal(?isY ?isY)
                     ]

[TemporalReasoning2:(?x tme:intEqual ?y) <-
                     (?x rdf:type tme:IntervalEntity)
                     (?y rdf:type tme:IntervalEntity)
                     (?x tme:begins ?beingsX)
                     (?beingsX rdf:type tme:XSDDateTime)
                     (?x tme:ends ?endsX)
                     (?endsX rdf:type tme:XSDDateTime)
                     (?y tme:begins ?beginsY)
                     (?beingsY rdf:type tme:XSDDateTime)
                     (?y tme:ends ?endsY)
                     (?endsY rdf:type tme:XSDDateTime)
                      equal(?beginsX ?beginsY)
                      equal(?endsX ?endxY)
                     ]


[TemporalReasoning3:(?x tme:inside ?y) <-
                     (?x rdf:type tme:InstantEntity)
                     (?y rdf:type tme:IntervalEntity)
                     (?x tme:begins ?beingsX)
                     (?x tme:is ?isX)
                     (?isX rdf:type tme:XSDDateTime)
                     (?y tme:begins ?beginsY)
                     (?beingsY rdf:type tme:XSDDateTime)
                     (?y tme:ends ?endsY)
                     (?endsY rdf:type tme:XSDDateTime)
                     greaterThan(?isX ?beginsY)
                     lessThan(?isX ?endsY)
                     ]


[TemporalReasoning4:(?x tme:intOverlap ?y) <-
                     (?x rdf:type tme:IntervalEntity)
                     (?y rdf:type tme:IntervalEntity)
                     (?x tme:begins ?beginsX)
                     (?beingsX rdf:type tme:XSDDateTime)
                     (?x tme:ends ?endsX)
                     (?endsX rdf:type tme:XSDDateTime)
                     (?y tme:begins ?beginsY)
                     (?beingsY rdf:type tme:XSDDateTime)
                     (?y tme:ends ?endsY)
                     (?endsY rdf:type tme:XSDDateTime)
                     lessThan(?beginsX ?beginsY)
                     greaterThan(?endsX ?beginsY)
                     lessThan(?endsX ?endsY)
                     ]
```

```
[TemporalReasoning5:(?x tme:intOverlap ?y) <-
                    (?x rdf:type tme:IntervalEntity)
                    (?y rdf:type tme:IntervalEntity)
                    (?x tme:begins ?beginsX)
                    (?beingsX rdf:type tme:XSDDateTime)
                    (?x tme:ends ?endsX)
                    (?endsX rdf:type tme:XSDDateTime)
                    (?y tme:begins ?beginsY)
                    (?beingsY rdf:type tme:XSDDateTime)
                    (?y tme:ends ?endsY)
                    (?endsY rdf:type tme:XSDDateTime)
                    greaterThan(?beginsX ?beginsY)
                    lessThan(?beginsX ?endsY)
                    greaterThan(?endsX ?endsY)
                    ]

[TemporalReasoning6:(?x tme:intContain ?y) <-
                    (?x rdf:type tme:IntervalEntity)
                    (?y rdf:type tme:IntervalEntity)
                    (?x tme:begins ?beginsX)
                    (?beingsX rdf:type tme:XSDDateTime)
                    (?x tme:ends ?endsX)
                    (?endsX rdf:type tme:XSDDateTime)
                    (?y tme:begins ?beginsY)
                    (?beingsY rdf:type tme:XSDDateTime)
                    (?y tme:ends ?endsY)
                    (?endsY rdf:type tme:XSDDateTime)
                    lessThan(?beginsX ?beginsY)
                    greaterThan(?endsX ?endsY)
                    ]

[TemporalReasoning7:(?x tme:intContain ?y) <-
                    (?x rdf:type tme:IntervalEntity)
                    (?y rdf:type tme:IntervalEntity)
                    (?x tme:begins ?beginsX)
                    (?beingsX rdf:type tme:XSDDateTime)
                    (?x tme:ends ?endsX)
                    (?endsX rdf:type tme:XSDDateTime)
                    (?y tme:begins ?beginsY)
                    (?beingsY rdf:type tme:XSDDateTime)
                    (?y tme:ends ?endsY)
                    (?endsY rdf:type tme:XSDDateTime)
                    lessThan(?beginsX ?beginsY)
                    equal(?endsX ?endsY)
                    ]

[TemporalReasoning8:(?x tme:intContain ?y) <-
                    (?x rdf:type tme:IntervalEntity)
                    (?y rdf:type tme:IntervalEntity)
                    (?x tme:begins ?beginsX)
                    (?beingsX rdf:type tme:XSDDateTime)
                    (?x tme:ends ?endsX)
                    (?endsX rdf:type tme:XSDDateTime)
                    (?y tme:begins ?beginsY)
                    (?beingsY rdf:type tme:XSDDateTime)
                    (?y tme:ends ?endsY)
                    (?endsY rdf:type tme:XSDDateTime)
                    equal(?beginsX ?beginsY)
                    greaterThan(?endsX ?endsY)
                    ]

[TemporalReasoning9:(?x tme:before ?y) <-
                    (?x rdf:type tme:IntervalEntity)
                    (?y rdf:type tme:IntervalEntity)
                    (?x tme:ends ?endsX)
                    (?endsX rdf:type tme:XSDDateTime)
                    (?y tme:begins ?beginsY)
                    (?beingsY rdf:type tme:XSDDateTime)
                    lessThan(?endsX ?beginsY)
                    ]

[TemporalReasoning10:(?x tme:before ?y) <-
                    (?x rdf:type tme:InstantEntity)
                    (?y rdf:type tme:IntervalEntity)
                    (?x tme:is ?isX)
```

```
                    (?isX rdf:type tme:XSDDateTime)
                    (?y tme:begins ?beginsY)
                    (?beingsY rdf:type tme:XSDDateTime)
                    lessThan(?isX ?beginsY)
                    ]

[TemporalReasoning11:(?x tme:before ?y) <-
                    (?x rdf:type tme:IntervalEntity)
                    (?y rdf:type tme:InstantEntity)
                    (?x tme:ends ?endsX)
                    (?endsX rdf:type tme:XSDDateTime)
                    (?y tme:is ?isY)
                    (?isY rdf:type tme:XSDDateTime)
                    lessThan(?endsX ?isY)
                    ]

[TemporalReasoning12:(?x tme:before ?y) <-
                    (?x rdf:type tme:InstantEntity)
                    (?y rdf:type tme:InstantEntity)
                    (?x tme:is ?isX)
                    (?isX rdf:type tme:XSDDateTime)
                    (?y tme:is ?isY)
                    (?isY rdf:type tme:XSDDateTime)
                    lessThan(?isX ?isY)
                    ]


# ----------- Example inference rules to define activity reasoning ------------------

[InferenceRule1: (?p per:isEngagedIn act:working) <-
                 (?p rdf:type per:person)
                 (?p per:hasContextualLocation ?lp)
                 (?lp rdf:type loc:ContextualLocation)
                 (?p per:hasWorkingPlace ?com)
                 (?com rdf:type loc:SpatialEntity)
                 (?com loc:hasLocation ?lc)
                 (?lc rdf:type loc:PremiseLocation)
                 (?lp loc:locatedIn ?lc)
                 ]


[InferenceRule2: (?p per:isEngagedIn act:traveling) <-
                 (?p rdf:type per:person)
                 (?p per:hasContextualLocation ?lp)
                 (?lp rdf:type loc:ContextualLocation)
                 (?p per:hasHome ?home)
                 (?home rdf:type loc:SpatialEntity)
                 (?home loc:hasLocation ?lh)
                 (?lh rdf:type loc:PremiseLocation)
                 (?lp loc:disconnectedWith ?lh)
                 (?p per:hasScheduledActivity ?act)
                 equal(?act act:traveling)
                 (?act act:HappenWhen ?dates)
                 (?dates tme:intContain tme:today)
                 ]
```